

by

James Russell Geraci

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degrees of

Master of Engineering

	1
nn	0
an	L.
	_

Bachelor of Science in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1999 June 19910

© James Russell Geraci, MCMXCIX. All rights reserved.

The author hereby grants to MIT permission to reproduce and distribute publicly paper and electronic copies of this thesis document in whole or in part, and to grant others the right to do so.

n

٨

Author	17-14-14-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1		······	
	Department of	Electrical En	gineering a	nd Computer Science May 21, 1999
Certified by	_			
				Dr. Tom Keim
\cap (Research Scientist
	4			Thesis Supervisor
Certified by		5		
				John G. Kassakian
			χ.	Professor
				Thesis Supervisor
Accepted by			_	
Accepted by	(7	Arthur C. Smith
	Chairman, I	Department <u>C</u>	<u>ommittee o</u>	n Graduate Students

42-Volt PowerNet System Management Using Multiplexed Remote Switching by

James Russell Geraci

Submitted to the Department of Electrical Engineering and Computer Science on May 21, 1999, in partial fulfillment of the requirements for the degrees of Master of Engineering

 and

Bachelor of Science in Electrical Engineering

Abstract

The main objective of this thesis is to explore techniques for using multiplexed remote switching in a 42/14 volt dual voltage automotive environment to perform bus energy management and other useful system functions. Achieving this objective involved first constructing a 42v/14v dual voltage automotive test facility. Then, designing and evaluating candidate algorithms for bus energy management in a dual-voltage electrical system using that test facility. The energy management algorithms explored in this thesis were designed to minimize the cost and equipment needed to implement the algorithms. This will allow future work to perform cost vs. performance gain analysis.

Thesis Supervisor: Dr. Tom Keim Title: Research Scientist

Thesis Supervisor: John G. Kassakian Title: Professor

Acknowledgments

I would like to thank Professor John Kassakian and Dr. Tom Jahns for originally entrusting me with this project. I would also like to thank Dr. Tom Keim for his insight and help with this project. I would like to thank my parents and brothers and sisters for their support. I would also like to thank Li Yu for her support and help with some of the most physically demanding parts of the project. Finally, I would like to thank Ed Lovelace and Dan Santos for their work in obtaining a drive motor for my project.

Contents

1	Intr	oducti	ion 1	.0
		1.0.1	Project Overview	10
2	Ene	ergy M	anagement Algorithms 1	.2
	2.1	Preser	nt Energy Management System	12
	2.2	42V/1	4V Energy Management System	13
		2.2.1	Bus Voltage Regulation	14
		2.2.2	Sophisticated Battery Model	14
		2.2.3	Artificial Intelligence	17
		2.2.4	Tested Energy Management Algorithms	18
			2.2.4.1 42v/14v Bus Regulation Algorithm	18
			2.2.4.2 Sophisticated Battery Model Algorithm	19
3	MI	Г Brea	dboard Facility 2	21
	3.1	Power	Delivery Systems	21
		3.1.1	The Breadboard Power Cabling	22
		3.1.2	Breadboard Batteries	22
		3.1.3	The Breadboard Alternator	23
		3.1.4	The Breadboard DC/DC Converter	24
	3.2	Power	Dissipating Systems	25
		3.2.1	Fixed Resistance Loads	26
		3.2.2	Speed Dependent Loads	26
	3.3	Contro	ol Systems	27

Contents

		3.3.1	PC Mas	ter Control System	27
			3.3.1.1	LabView File Input	28
			3.3.1.2	CAN Bus I/O	30
			3.3.1.3	Electromechanical Valve I/O	31
			3.3.1.4	Alternator Speed Control I/O	31
			3.3.1.5	User Interface Related Activities	31
			3.3.1.6	LabView File Output	32
		3.3.2	The CA	N bus and the C167CR \ldots	32
			3.3.2.1	The CAN Bus	36
		3.3.3	Load No	odes	36
		3.3.4	Energy	Management Node	37
		3.3.5	Serial to	O CAN Router Node	39
		3.3.6	Data Co	ollection Module	41
		3.3.7	PC Inpu	ıt Files	42
4	Test	t Proc	edure		46
	4.1	Design	n an Ener	gy Management Algorithm	46
		4.1.1	Selecting	g a Drivecycles	46
		4.1.2	Loadcyc	eles	47
	4.2	Test F	rocedure		48
5	Res	ults ar	nd Conc	lusion	49
A	Con	nplete	Sophist	icated Energy Management Algorithm	51
B	Bre	adboa	rd Code		56
	B.1	Organ	ization		56
	B.2	14V B	us CAN	Node 1	56

Contents

B.3	14V Bus CAN Node 2	58
B.4	14V Bus CAN Node 3	59
B.5	42V Bus CAN Node 1	60
B.6	42V Bus CAN Node 2	61
B.7	42V Bus CAN Node 3	62
B.8	CAN Router	63
B.9	Data Acquisition Node	64
B.10	DC/DC Converter Node	64
B.11	Saber to Breadboard Converter Code	65
B.12	Breadboard Loads	65

1.1	Dual Voltage Architecture with Communications Bus	10
2.1	Typical Voltage-time Discharge Curves of Lead Acid Cells	13
2.2	Battery State of Charge Partitioning used for this Thesis	14
2.3	Decision chart based on state of charge	16
2.4	Regulation Curve for DC/DC converter	18
2.5	40v alternator Current vs. RPM Characteristic	19
3.1	Diagram of MIT Breadboard Facility	22
3.2	40V Bosch Alternator Wiring Diagram	23
3.3	Digital Input of the MIT Breadboard DC/DC Converter	25
3.4	Circuit Diagram of BTS660P Smart Switch Board	27
3.5	Circuit Diagram of BTS550P Smart Switch Board	28
3.6	A few lines from a breadboard input file	28
3.7	LabView 'G' code that parses breadboard input files $\ldots \ldots \ldots \ldots \ldots \ldots$	30
3.8	C167CR Startup Code	34
3.9	Memory Map of Phytec KitCON-167 used in Breadboard Facility	34
3.10	Assembly Code that allows External Memory Bus Accesses	35
3.11	Loop Code for C167CR	35
3.12	CAN Message Object Regsiters and Memory Locations	37
3.13	Format of Serial Message	40
3.14	Precision Absolute Value Circuit with Direction SubCircuit	42
3.15	The LabView Breadboard Interface	44

3.16	The major communicating subsystems	45
5.1	Battery Voltages vs Time	49
A.1	Decisions made when 12v Battery is in the "Dangerous Overcharge" Region	51
A.2	Decisions made when 12v Battery is in the "Acceptable Overcharge" Region	52
A.3	Decisions made when 12v Battery is in the "Ideal Operation" Region	53
A.4	Decisions made when 12v Battery is in the "Acceptable Undercharge" Region	54
A.5	Decisions made when 12v Battery is in the "Dire Undercharge" Region	55

3.1	PacTorq Motor to SC756 Motor Driver Wiring Connections	24
3.2	Fixed Resistance Breadboard Loads	29
3.3	14v Bus CAN Messages	38
3.4	42v Bus CAN Messages	43
4.1	Variables Used in Car Velocity to Alternator Conversion	47

Chapter 1 Introduction

1.0.1 **Project Overview**

The objective of this thesis project was to explore techniques for using multiplexed remote switching in a dual-voltage system to perform bus energy management and other useful system functions. "Multiplexed remote switching" is a term used to describe the ability of an in-car computer network to control the state of various loads within the automobile. Such a system would require a data network, several microcontrollers, and switches whose state can be controlled by the microcontrollers. Because of the ever increasing amout of wiring in automobiles, the next generation automobile electrical system will have such a remote switching network installed. Figure 1.1 shows the main parts one possible topology for the next generation automotive electrical system. It is a 42/14 volt unidirectional DC/DC converter based automotive electrical system.

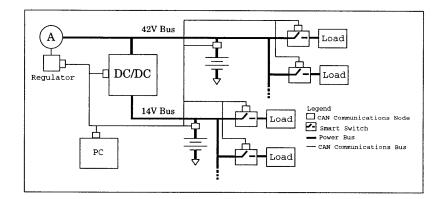


Figure 1.1: Dual Voltage Architecture with Communications Bus

In this dual voltage environment there are two voltage busses, a 42 volt and a 14 volt bus. Loads are attached to each bus and their on/off state is controlled by a microcontroller controlled switch. An example of a 42 volt load would be a front windshield heater. A 14 volt bus load might be the dome light that turns on when the car doors are opened. A complete list of loads used for this thesis can be found in Table 3.2.1. There are three sources of electrical power in the system of Figure 1.1. The first is the alternator and the others are the batteries. When the gasoline engine is running, it turns the alternator which converts the mechanical power of the engine into the electrical power used to supply the electrical system of the car. The batteries have different functions depending on if the car is running (key-on) or not(key-off). When the car is running, the two batteries perform a load leveling function. They provide power to their respective busses when the total demand for power on a bus exceeds the amount that is being provided to that bus by the alternator. When the car is off, each battery has a different function. The 42 volt battery's function is to start the car. The 14 volt battery's function is to ensure that the key-off loads have power during the entire time the car is off. The DC/DC converter acts as a regulated valve controlling power flow between the two busses.

If size, weight, and money were not an issue, the alternator should be sized so that it would be able to provide enough power so that there would be no possible combination of loads which could drain the batteries. Because of physical and economic limitations, however, such an alternator is not obtainable. Furthermore, such an alternator might not be the most desirable alternative. Due to the start and stop nature of automobile driving, there are times when the car batteries are being drained and times when they are being charged. The important thing is that the change in state of charge of each battery over the complete drive cycle is zero or positive. If it were possible to intelligently control the flow of charge between the two batteries so that no net charge is lost by either battery over a given drive cycle, it would be possible to size the alternator so that it would not have to provide enough power to keep both batteries fully charged at all times. This method of intelligently controlling the flow of energy throughout the automobile is called active energy management. Such an energy management system would allow the use of a smaller alternator and therefore reduce the weight and cost of the automobile.

It is highly likely that the next generation of automobile electrical system will include a multiplexed remote switching network. If it does include such a network, then the system will have the necessary communications and control elements to perform not only the communications necessary for an energy management algorithm to work but also to perform the computations necessary to make intelligent decisions based on the state of the automobile's electrical system. It is the purpose of this research to use a multiplexed remote switching network to investigate the performance of a number of energy management algorithms.

Chapter 2 Energy Management Algorithms

Energy management involves the estimation of energy consumption, proper sizing of equipment to meet this estimate, and proper operation of the equipment [1]. Energy management algorithms are a way to control the flow of energy throughout an automobile's electrical system. All energy management algorithms take in information about the system's state in order to try to determine the state of charge of the batteries. State of charge is a term often used to refer to the amount of work that the battery can do given an instantaneous set of environmental parameters¹. In addition, each algorithm can be customized to not only take into account information about the state of the system but also take into account safety information and preferences which might be of most benefit to the vehicle operator. For example, in the case of the energy management algorithms developed for this thesis, a strong preference was given to the operator being able to start his car. The system then combines the physical information and the preference information and uses that information to appropriately modify the state of the system's energy sources and sinks.

2.1 Present Energy Management System

Energy management algorithms are not new to the automobile industry. Today's automobile employs a simple yet effective energy management algorithm. It uses a voltage sensor that has a temperature compensated output voltage to measure the battery's voltage and uses this information to control the excitation of the alternator field winding, and thus the amount of power that the alternator will deliver to the system. This energy management algorithm uses curve A from Figure 2.1 as it's battery model [2].

Curve A in Figure 2.1 is a graph of battery cell voltage versus time for a battery which is slowly being drained at a constant current. Because batteries are rated in amp·hours, if the total charge leaving a battery is measured and the initial state of the battery is known, the state of charge

¹Not all algorithms actually calculate a state of charge. Most take action based on physical indicators necessary to compute the state of charge, but do not actually compute the state of charge itself

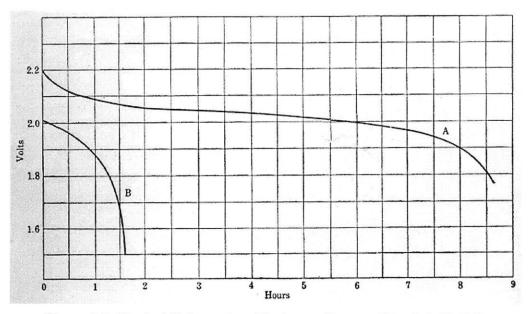


Figure 2.1: Typical Voltage-time Discharge Curves of Lead Acid Cells

of the battery can be computed. By using this graph, a relationship between the voltage of the battery and the battery's state of charge can be made. The present system of observing the bus voltage and then modifying the alternator excitation accordingly is simply trying to use the voltage information to make a guess at how much charge has been removed from the battery during a drive cycle. This algorithm does not compute a number for the state of charge, but simply reacts to the voltage which is an indicator of the state of charge of the battery.

2.2 42V/14V Energy Management System

The 42V/14V electrical system will also employ an energy management algorithm; however, the fact that there are now two batteries makes the control of the system more complex and the possible benefits of having a good energy management algorithm greater. This thesis three main levels of sophistication for an energy management algorithm.

- 1. Bus Voltage Regulation
- 2. Sophisticated Battery Model
- 3. Artificial Intelligence

2.2.1 Bus Voltage Regulation

Bus voltage regulation is the 42V/14V extension of the present day energy management algorithm. It employs a temperature compensated voltage sensor on the outputs of the DC/DC converter and the 42V alternator to measure the voltage on each bus and then uses curve A in Figure 2.1 to infer the state of charge of each battery. It has the advantage that it can be easily implemented and can be expected to maintain battery charge for both batteries about as well as today's highly satisfactory system.

2.2.2 Sophisticated Battery Model

The second level of sophistication employs a more sophisticated battery model than the bus voltage regulation level. This level employs state of charge explicitly rather than implicitly through bus voltage. By reasoning about battery state of charge directly, it becomes possible to make more intelligent decisions about how to control the states of the energy sources and sinks on the network and thus develop an energy management algorithm. One way to use state of charge information to help develop an energy management algorithm is to first break each battery's state of charge into a number of different regions and then make decisions based on which region each battery is in at any given time. An example of how a battery's state of charge might be decomposed into different regions is shown in Figure 2.2.

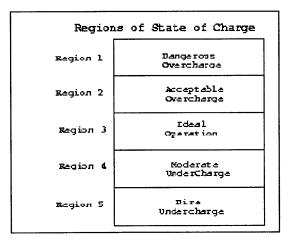


Figure 2.2: Battery State of Charge Partitioning used for this Thesis

Figure 2.2 shows the battery state of charge broken into 5 different regions. The exact place in the state of charge continuum where each of the regions starts and stops have not yet been standardized; however, for the purpose of this thesis, the following divisions were created:

- Region 1: Dangerous Overcharge $\rightarrow 115\% \leq \text{SOC}$
- Region 2: Acceptable Overcharge $\rightarrow 105\% \leq \text{SOC} < 115\%$
- Region 3: Ideal Operation $\rightarrow 90\% \leq \text{SOC} < 105\%$
- Region 4: Moderate Undercharge $\rightarrow 50\% \leq \text{SOC} < 90\%$
- Region 5: Dire Undercharge \rightarrow SOC < 50%

Figure 2.3 shows the 5x5 decision matrix which graphically displays the 25 different possible regions into which the states of charge of both batteries may fall. The numbers on each edge correspond to the state of charge regions in Figure 2.3.

A few examples of possible decisions based solely on the state of charge of the batteries are written in the boxes in Figure 2.3. If both batteries are in a dangerous state of overcharge, then the algorithm would turn off the DC/DC converter, decrease the alternator field winding excitation (possibly turning it off), and turn on select high power loads on both the 42v bus and the 14v bus. These actions would immediately cut off power flow into the 12v battery, so it would begin to discharge. It would also allow the 36v battery to begin discharging as rapidly as possible. This kind of situation would not occur in the voltage regulation energy management system unless something had gone wrong with the voltage regulators, so actions taken during this mode of operation can be seen as a sort of a safety device.

Another situation the system might get in is if both batteries are in a dire state of undercharge. This situation might occur if, over a period of time, both batteries are drained and not returned to a full state of charge after each drive cycle. In such a situation there might be the possibility of recharging one of the batteries. This is where the engineer must make a decision as to what action would best serve the customer. The system could either be designed to let the DC/DC converter try to regulate the 14v bus and thus hopefully save the 12v battery, or it could be designed to shut the DC/DC converter off and hopefully save the 36v battery.

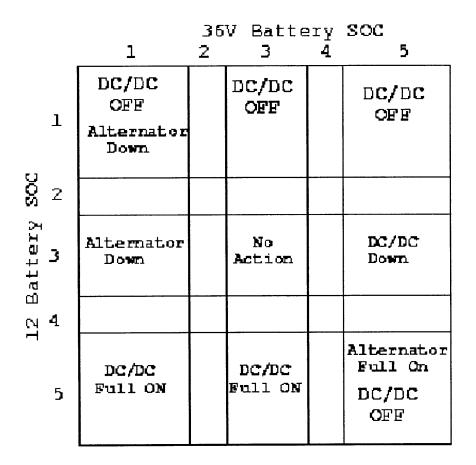


Figure 2.3: Decision chart based on state of charge

If, in addition to the state of charge information, the current on each battery were known, then even more informed decisions could be made. For example, if both batteries were in a region of acceptable undercharge, but the 12v battery was draining, while the 36v battery was being charged, the system could be designed so that the DC/DC converter would pass more current to the 12v battery without causing the 36v battery to drain. This would keep the 36v battery in an acceptable region of charge and it would either minimize the rate at which the 12v battery discharged, thus extending the life of the 12v battery, or it might allow the 12v battery to begin charging. It might even be possible for both batteries to charge. For example, if the 36v battery were charging at a rate of 6 amps, and the 12v battery were discharging at a rate of 5 amps, it might be possible to control the DC/DC converter so that the 36v battery would charge at a rate of 3 amps and the 12v battery would charge at a rate of 4 amps.

The benefit of the sophisticated battery model energy management algorithm, over the simple voltage regulation algorithm, is that the designer of the electrical system has more flexibility to dictate how the system responds to different loading states. Because this algorithm can limit the amount of current delivered by the DC/DC converter, it is possible to charge the 12v battery at a rate that is less than the converter's maximum current delivery capability. With reduced output, the current drawn from the 42V bus by the converter is reduced. This current can instead go to the 36V battery thus reducing its rate of discharge and possibly even allowing it to charge. Therefore, the situation could exist where both batteries are charging, albiet very slowly, instead of in the voltage regulation case where only one battery is charging rapidly and the other is draining because it is feeding the charging battery.

2.2.3 Artificial Intelligence

The decisions made by the energy management algorithm become the most helpful when the system is aware of the physical environment around the car and can possibly learn the operator's driving habits. Such a system might be aware of the date, the time of day, and the outside temperature. It could be made aware of the weather forecast by having it automatically dial into the weather service each night so it could adjust how it behaves for the following day. It could also be plugged into a GPS system. If it then knew its starting point and its finish it could calculate the amount of time that it would be driving and possibly the type of driving (in city or country) that it would be doing. This information could have a significant impact on the way that energy is managed in the system. Take again, for example, the situation where both batteries are in an acceptable state of undercharge and the 12v battery is discharging and the 42v battery is charging. If, the car knew that it was going to be doing a short drive and that the 12v battery wasn't discharging too rapidly, it might choose to decrease the output of the DC/DC converter so that the 12v battery drained a little more rapidly, but the 36v battery would charge more rapidly and might possibly move into a region of ideal operation.

Finally, the decision on how to control the DC/DC converter would change once again if the car were able to learn the driver's driving habits. If, for example, it were Friday at 6PM and the car knew that the driver always went to his cabin for the weekend, and that the driver just let his car sit over the weekend, the car would want to try to maximize the charge on the 12v battery by increasing the output of the DC/DC converter so that it would be able to power all of its key-off loads for the weekend.

2.2.4 Tested Energy Management Algorithms

For the purpose of this thesis both the 42v/14v bus regulation algorithm and a sophisticated battery model algorithm were tested. Information about the load cycles, drive cycles, and physical test facilities used to test these energy management algorithms can be found in Chapter 4. The sophisticated battery model algorithm was limited to controlling only the state of the DC/DC converter.

2.2.4.1 42v/14v Bus Regulation Algorithm

The 42v/14v bus regulation algorithm which was tested simply used the voltage regulators on the DC/DC converter and the alternator to control the flow of power throughout the system. Figure 2.4 shows the regulation characteristic of the DC/DC converter. This curve means that the DC/DC converter will try to deliver it's maximum current of 68 amps anytime the voltage on the 12v battery drops below 13.8 volts. Figure 2.5 shows the alternator's regulation characteristic. The alternator is set to regulate its output to 40 volts, and it can deliver up to 90 amps in order to maintain a 40 volt output voltage.

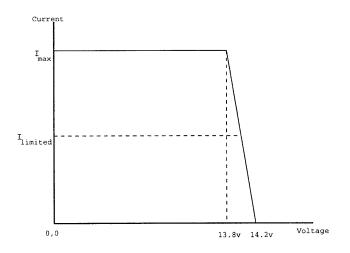


Figure 2.4: Regulation Curve for DC/DC converter

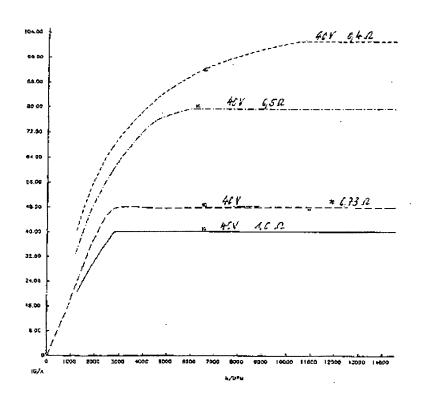


Figure 2.5: 40v alternator Current vs. RPM Characteristic

2.2.4.2 Sophisticated Battery Model Algorithm

The sophisticated battery model algorithm which was designed was based on a coulomb counting algorithm. The amount of current coming out of each of the batteries was measured about once every second and its integral was computed. This value was then used to compute the percent state of charge according to Formula 2.1.

State of Charge =
$$\frac{\text{(Initial Amp \cdot hours)} - (\text{Amp \cdot hours used)}}{\text{(Initial Amp \cdot hours)}}$$
(2.1)

Once the state of charge for each battery was calculated², the system's present operating region in Figure 2.3 was determined. From there, current information was used to make a final decision about the state of the DC/DC converter. A complete enumeration of all possible decisions can be found in Appendix A

Long-term inaccuracies in the discrete approximation of the total change in charge of a battery will result in the true state of charge diverging over time from the state of charge calculated the present test facilities data collection equipment. Even if it were possible to count every coulomb entering and leaving the battery, the calculated state of charge and the true state of charge would diverge due to internal self-discharge mechanisms. Over a long period of time, any control algorithm that computes battery state of charge soley on equation 2.1 would need to be supplemented by additional information sensitive to actual state of charge, for example, voltage and temperature. For the purpose of this thesis, however, the rate of divergence between calculated and actual state of charge should be slow enough to permit meaningful observations.

²State of charge will be used to mean percent state of charge from now on.

Chapter 3 MIT Breadboard Facility

In order to validate the energy management algorithms that were discussed in Chapter 2, it was necessary to construct physical test facilities on which those tests should be conducted. The facility that was to be constructed had to be an easily controllable and modifiable electrical equivalant of the 42V/14V unidirectional DC/DC converter architecture from Figure 1.1. the facility can be broken down 3 major parts.

1. Power Delivery Systems: Section 3.1

The Breadboard Power Cabling: Section 3.1.1

The Breadboard Batteries: Section 3.1.2

The Breadboard Alternator and Support Hardware: Section 3.1.3

The Breadboard DC/DC Converter: Section 3.1.4

2. Power Dissipating Systems: Section 3.2 Fixed Resistance Loads: Section 3.2.1

Speed Dependent Loads: Section 3.2.2

3. Control Systems: Section 3.3

PC Master Control System: Section 3.3.1

The C167CR: Section 3.3.2

The CAN Bus: Section 3.3.2.1

Data Collection System: Section 3.3.6

Software to generate PC input files: Section 3.3.7

3.1 Power Delivery Systems

The breadboard power delivery system is made up of all sources of power and the physical cabling necessary to deliver that power to the systems loads. This includes the batteries, the alternator

and its support equipment, the DC/DC converter and the cables necessary to deliver power to the loads.

3.1.1 The Breadboard Power Cabling

A diagram of the power cabling for the MIT breadboard facility can be seen in Figure 3.1.

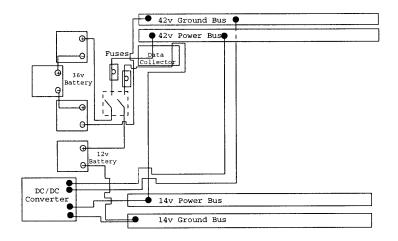


Figure 3.1: Diagram of MIT Breadboard Facility

Each power and ground bus was implemented by an aluminum rail. The two power busses are located on opposite sides of the breadboard facility. Leads from loads can be screwed to each of the rails. There are two separate ground rails. These represent different local grounds that might occur in an automobile. They are connected together by a pair of 4 AWG cable. This pair of cable performs the same function as that of a chasis in an automobile.

3.1.2 Breadboard Batteries

The 36V battery was made up of 3 AC Delco Professional Freedom Car and Truck 58-5YR batteries connected in series. They have a reserve capacity¹ of 70 minutes. The 12V battery was

¹Reserve Capacity [3] is the ability of the battery to maintain a cell voltage of 1.75V or greater at a discharge rate of 25 amps.

an AC Delco Professional Freedom Car and Truck 65-7YR battery. It has a reserve capacity of 160 minutes.

3.1.3 The Breadboard Alternator

The alternator used to provide power to the network was a 40V Bosch alternator that was given to the MIT Constorium for Advanced Automotive Electrical And Electronic Equipment by Paul Nicastri of Ford. The alternator can supply 50 amps at idle and 90 amps at higher rpm. Thus the alternator can supply a maximum of 2000 watts at idle and 3600 watts at higher rpm. Its output current vs. rpm characteristic can be seen in Figure 2.5. The appropriate wiring diagram for the alternator can be seen in Figure 3.2.

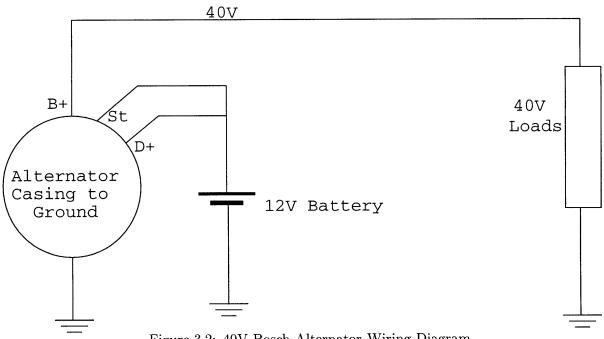


Figure 3.2: 40V Bosch Alternator Wiring Diagram

In a conventional automobile, the alternator is spun by the car's engine. It is geared at a ratio of approximately 3 alternator rotations for every one engine rotation. The situation is the same with the breadboard facility. The alternator was controlled by an 18hp 13.4kW Pacific Scientific PacTorq Brushless P.M. Servomotor. The servomotor and the alternator were geared so that one rotation of the servomotor produces about 3 rotations of the alternator. The speed of the motor was controlled by a Pacific Scientific 756 ServoController. The appropriate wiring of the 756 ServoController to

the PacTorq servomotor can be seen in Table 3.1. The controller is controlled through its serial port, and for testing purposes, it is being software limited by its control program, 'PacTorq.bas'², to spinning the PacTorq motor to 3500rpm. If this limit is exceeded, the motor stops all motion and cannot move again until it is reprogrammed.

Power Connections				
PacTorq Motor Connection Label	SC756 Drive Connection Label			
T_1	Т			
T_2	R			
T_3	S			
Resolver	Connections			
Pactorq Motor Connection Number	SC756 Drive J51 Connection Number			
1	4			
2	3			
3	2			
4	1			
NONE	5			
5	6			
6	7			
7	NONE			
8	8			
9	9			
10	NONE			

Table 3.1: PacTorq Motor to SC756 Motor Driver Wiring Connections

3.1.4 The Breadboard DC/DC Converter

The breadboard's DC/DC converter is a unidirectional converter that is capable of delivering up to 68 amps to the 14v bus. It's regulation characteristic can be seen in Figure 2.4. The DC/DC converter can be controlled to deliver an amount of current less than its instantaneous maximum deliverable power. An example of this can be seen in Figure 2.4. In Figure 2.4 the converter can supply I_{max} but it can also supply any amount of current less than I_{max} like $I_{limited}$ for example. The converter, however, cannot be controlled to deliver an amount of current greater than its regulation characteristic will allow. For example, if the 14V bus were at 14.0V (it is regulated to 14.2v) then the maximum amount of current that the converter could deliver is 34 amps. It cannot

²'PacTorq.bas' can be found in Appendix B

be controlled in any way to deliver more than 34 amps, but it can be controlled to deliver any amount of current less than 34 amps.

The current limit of the DC/DC converter can be set by changing the value that appears on its 8-bit input seen in Figure 3.3.

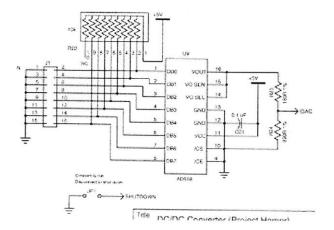


Figure 3.3: Digital Input of the MIT Breadboard DC/DC Converter

Each input pin of the AD558 A/D converter has a pull up resistor. The pin can be brought to logical low by first connecting an open drain configured transistor to the resistor and then activating that transistor. The converter is at maximum current when all of the pins are high, and it is at zero current when all the pins are low. Pin DB0 on the AD558 is the LSB. The on/off state of the converter is controlled by a separate pin. The converter will turn on when this pin is connected to ground.

3.2 Power Dissipating Systems

By the year 2005 some automobiles will have an average electrical load of over 2500 watts [4]. The electrical loads for the breadboard were selected in order to allow loading in excess of 2500 watts. The loads that were selected for the breadboard facility can dissipate a total of about 4100 watts. This is well above the maximum alternator output of 3600 watts at high alternator rpm. Therefore, because the batteries must be used, an energy management algorithm is relevant.

In the case of the breadboard facility, loads can be broken down into two different categories. The first type of load is a fixed resistance load, and the second type of load is a speed-dependent load. For the MIT breadboard facility 11 different fixed resistance loads were selected and implemented as CAN enabled smart switch controlled loads. The electromechanical valve system was the only speed-dependent load enabled on the breadboard. It is discussed in Section 3.2.2.

3.2.1 Fixed Resistance Loads

The loads that were selected as fixed resistance loads are shown in Table 3.2.1. The resistors were held in aluminum mounts and power flow to the resistors was controlled by a microcontroller controller power MOSFET. The Siemen's BTS550P was used to switch on and off loads on the 14V bus, and the Siemen's BTS660P was used to control loads on the 40V bus. Each MOSFET provides as an output on one of its pins a current that is proportional to the amount of current flowing through its channel. The MOSFETs were mounted to custom designed boards. Also mounted to each board was a LM317 voltage regulator that was used to provide power to the CAN microcontroller that was controlling the state of the MOSFET via instructions it was receiving over the CAN bus³ A circuit diagram for the BTS660P's board can be seen in Figure 3.4, and a circuit diagram for the BTS550P's board can be seen in Figure 3.5.

3.2.2 Speed Dependent Loads

The electromechanical valve system was the only speed-dependent load enabled on the breadboard. It was implemented using a Hewlett Packard 6050A 1800Watt Programmable Load that was configured to draw a current proportional to the speed of the alternator. The amount of current it demanded was varied with alternator speed according to Equation 3.1. It has a minimum demand of 9 amps at idle (alternator speed of 1800 rpm) and a maximum of 45 amps at higher speeds (alternator speed of 6000 rpm or more). The HP 6050A received control commands over a GPIB bus.

³See Section 3.3.2 for a detailed description of the CAN bus.

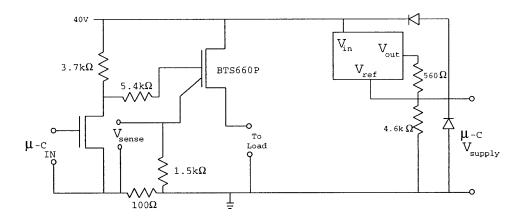


Figure 3.4: Circuit Diagram of BTS660P Smart Switch Board

$$I_{demanded} = \frac{9}{350} \text{Motor}_{rpm} - 6.425 \tag{3.1}$$

3.3 Control Systems

3.3.1 PC Master Control System

Because the breadboard facility cannot be driven, a method of simulating driving had to be created. This virtual driver was implemented using LabView 5.0. The virtual driver was coded in LabView's multithreaded 'G' graphical programming language and run on a 200 MHz Pentium PC running Windows95. Figure 3.15 shows the final PC interface for the facility. The virtual driver had to be able to turn on and off fixed resistance loads, control the amount of current drawn by the DC/DC converter, control the speed of the alternator, and collect information about the state of the system. A subprogram was written to control each of these functions, and these subprograms were combined toghter in the file "testcircuit2.vi." The major subprograms⁴ are shown in Figure 3.16. The current drawn by the DC/DC converter is controlled by 'EMValve.vi.' The speed of the alternator is controlled by 'PACSCIBYTE.vi', Information going to and received from the CAN bus is controlled by 'SerialController.vi.' Information is sent through the CAN bus to the PC, so

⁴Programs and subprograms are called 'VIs' in LabView

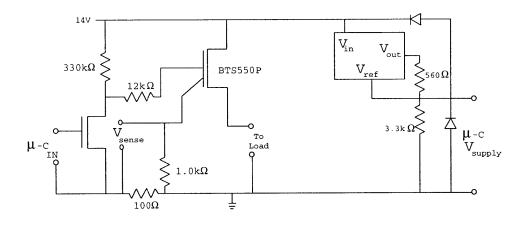


Figure 3.5: Circuit Diagram of BTS550P Smart Switch Board

the CAN bus is the means of collection of information about the state of the system.

3.3.1.1 LabView File Input

The virtual driver itself is implemented in 'fileinputtest2.vi'. Fileinputtest2.vi reads in a specially formatted file into a gian 2D array and then converts the information in the 2D array into information that in the appropriate 'vi' can use to create electrical events on the breadboard facility. This file is generated by a custom Java program that is described in Section 3.3.7. A few lines from one of these files can be seen in Figure 3.3.1.1.

Figure 3.6: A few lines from a breadboard input file

Fileinputtest2.vi parses each line of the breadboard input file into a number of different tokens. The information portion of each token is written to a global variable that has been designated as a holder of that token's information. This global variable is, in turn, read by the appropriate subvi. For example, take the line from Figure 3.3.1.1 that starts with "!54". This line would be broken into 4 different tokens. The first token starts with a '!'. This tells the file input subprogram that

Breadboard Loads						
	14V Bus Loads					
Load Name	Saber Name	Wattage	Current	Resistance		
Power Door Locks	sdr_locks	88	6.0	2.4		
Seat and Door Module	sdr_seat_adjust	13	1	15		
Turn Lights	sdr_turn	111	7.9	1.8		
ABS	sdr_abs_tc	324	23	0.6		
Brake Loads	sdr_brakes	146	10.5	1.3		
42V Bus Loads						
Rear Seat Heater	sdr_rear_seat_htrs	180	4.29	9.78		
Air Pump	sdr_emissions	480	11.4	3.7		
Heated Windshield	sdr_windshield	700	16.7	2.5		

Table 3.2: Fixed Resistance Breadboard Loads

the following information is the time offset, in seconds, since the start of the test. It is written to the global variable "Time Counter Global." The next token, '?', tells fileinputtest2.vi that this information is the new speed, in rpm, of the PacSci Servomotor. Information following a '?' is written to global variable "RUNSPEED." The alternator rotates at 3 times the value in this global variable. The third token "42+" tells fileinputtest2.vi that this information is the new amount of current to be demanded by the programmable load. It amount of current to be demanded is written to global variable "E&M valve current demand." In this case the amount of current to be demaded is 14 amps. The fourth token, "#" indicates that the following data is a CAN message. It is written to global variable "CAN write buffer." The final token, "//" tells fileinputtest2.vi that this is the end of the line and that it should proceed to the start of the next line. It is important to note that not all lines will have all tokens, and, therefore, the length of the lines in the input file may vary. The line starting with "57" only contains 3 tokens compared with the 5 of the line starting with "154". This helps greatly reduce the size of the breadboard input file and this in turn greatly improves the performance of the entire system because it allows better use of the host PC's processing power. The LabView code that reads in the breadboard input file and parses it can be seen in Figure 3.7. The code consists of two large while loops and several inner condition statements. Every time through the inner loop consists of reading in and testing a token, i.e. reading in a single column element from a row and sending the information in the column element to the appropriate global variable. Execution of the outer loop corresponds to changing to a new row.

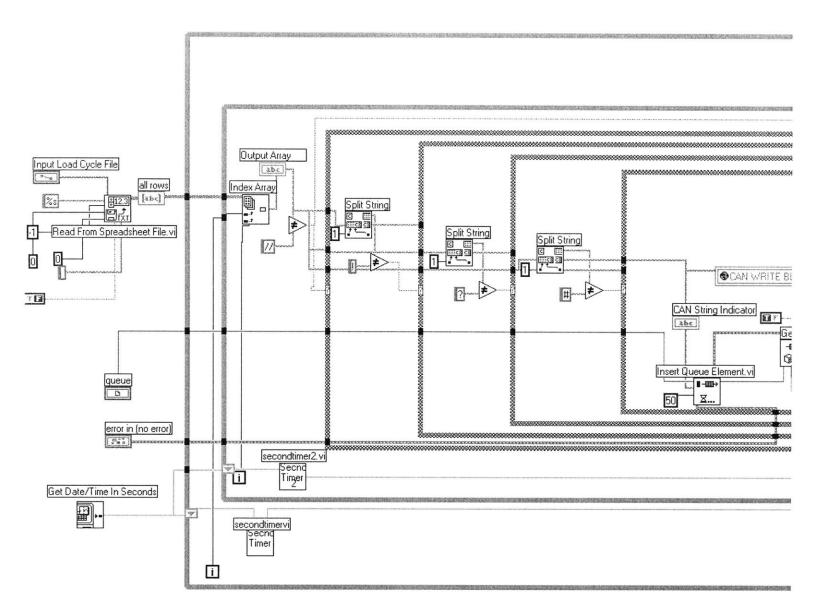


Figure 3.7: LabView 'G' code that parses breadboard input files

3.3.1.2 CAN Bus I/O

Unfortunately, there is no known CAN interface to LabView. In order to use a PC card that will allow the system to connect directly to the CAN bus, LabView would have to call a Windows dynamic linked library function. LabView is implemented so that when it calls a Windows dll, LabView stops all threads from executing until that dll function call is complete. This means that every time the system wants to watch activity on the CAN bus, or receive a piece of information from the CAN bus, LabView would have to stop all threads of execution and wait. If the CAN bus were accessed more than a few times a second, the system could quickly get bogged down. LabView does, however, have native serial port accessing methods, and it has serial port support though

its native VISA⁵ support. It was decided then, that the PC would be connected to the CAN bus through a serial router. Presently, this serial router only operates at 9600 baud; however, the serial router can be operated at baud rates up to 625KBaud. The operation of the router is described in Section 3.3.5.

3.3.1.3 Electromechanical Valve I/O

The electromechanical valve I/O subprogram was also implemented in VISA, and it's software is almost identical to that of the CAN I/O subprogram except for the fact that it only transmits data and never requests feedback from the programmable load. The programmable load has 3 different 600 Watt channels that can be controlled together to give up to 1800 Watts. The electromechanical valve I/O subprogram divides the demand between the three channels evenly. Each channel never demands more than 15 amps individually.

3.3.1.4 Alternator Speed Control I/O

The alternator speed was controlled through communications port 1 (COM1) on the PC. It's interface program was written using LabView's VISA modules so it should run on Windows NT as well as Windows 95. It operates by sending a string through the serial port to the servocontroller that was controlling the speed of the alternator. For example, if it was desired to have the alternator spin at 900 rpm, then the string "00900" (plus a carriage return) was written to COM port 1. There are always 2 leading zeros because LabView uses one and the servocontroller uses the second one to create an interrupt to which it will respond. Therefore, the third value 9 is the first value read in by the servocontroller. The string "00900" will cause the servomotor to spin at 900 rpm. This means that the alternator is spinning at 2700 rpm.

3.3.1.5 User Interface Related Activities

The user interface is the lowest priority subprogram in LabView. Under heavy loading situations LabView will often not update the interface right away. This can give the appearance of a delay in the network; however, this is not the case. It is only LabView trying to make sure that all I/O subprograms operate properly even at the expense of the user interface. This portion of the

⁵VISA is an interface which allows you to access all of the PC's I/O ports in an identical fashion through generic Read/Write commands. Therefore, it is possible to use almost the same code to access a GPIB port as it is to access a serial port.

program is also responsible for writing collected data to the hard disk. The data that is written is battery voltage for each battery, time, and motor rpm.

3.3.1.6 LabView File Output

LabView takes bus voltage, alternator shaft speed and battery current information and writes it to an output file. By default, the file is named "output.txt" and it located in the root directory of the "d:" drive of the PC that was used. The output file is a tab delimited file. The columns in the file represent time, alternator shaft speed, 42V bus voltage, 14V bus voltage, 42V bus current, 14V bus current, state of DC/DC converter.

3.3.2 The CAN bus and the C167CR

One of the features of the next generation of automobile electrical system may very well be some type of multiplexed data network that will control the state of the loads. The breadboard facility implements this feature in the form of a CAN network. CAN is a Bosch networking protocol which was developed in the late 1980's for use in the automotive industry. CAN is an acronym which stands for Controller Area Network. A complete discussion of the specifics of the CAN network protocol can be found in the book "CAN System Engineering: From Theory to Practical Applications" [5]. CAN is a standard for transmitting messages, and the exact hardware implementation might vary between vendors. For the purpose of this thesis it is important to understand the Siemens C167CR microcontroller, and how Siemens implements the CAN protocol in this controller.

The C167CR microcontroller is a 16-bit microcontroller. The CPU is able to operate at clock speeds of up to 20 MHz. One of the major applications for microcontrollers is data collection and real time control of external systems. To better achieve this goal, there is an on chip peripheral subsystem that operates independent of the CPU core. This peripheral subsystem is connected with the CPU via a complex system of interrupts. If the peripheral needs the CPU to perform some task, the peripheral requests the attention of the CPU by generating an interrupt. Ingeneral, the peripheral will not do anything while it is waiting for its interrupt request to be serviced. The peripheral subsystem contains 9 different peripherals all of which operate independent of the other peripherals and the CPU. Four peripherals are used in this thesis. They are the A/D converter, the General Purpose Timer Units, the Asynchronous Serial Channel, and the CAN-Module.

The C167CRs that make up the breadboard facilities CAN come in four main varieties.

- 1. Load Nodes
- 2. DC/DC Converter Controller Node
- 3. Energy Management Node
- 4. Serial to CAN Router Node

The software that controls each of these nodes is made up of a 'mainXYZ.asm'⁶ object file that is linked to several other object files that control one of the on chip peripherals. A full list of each node and the software that makes up the node can be found in Appendix B.

The files are assembled togther using a DOS batch file entitled 'compXYZ.bat' where XYZ is a unique alphanumeric identifier for each node. 'CompXYZ.bat' first assembles all of the necessary assembly files. It then proceeds to link these files and locate them, and then turn the output of the locater⁷ into an Intel hex formatted file. Intel hex is the file format required by the KitCON-167 board. All Intel hex formatted files end in '.hex'. These files can be downloaded to the KitCON-167 boards via the program 'Flasht.exe'. Download of an Intel hex formatted program to one of the KitCON-167 boards is done by first connecting the KitCON-167 board to the COM1 port of the PC. Then, 'flasht' must be typed and entered from a DOS command prompt in the directory that contains the hex file that should be downloaded. The 'Flasht.exe' program will only work properly if it is in the Windows95 path⁸. 'Flasht.exe' does not work under Windows NT.

A microcontroller differs from a PC in that the microcontroller does not come with a preprogrammed boot ROM or BIOS. The information in the PC's BIOS tells the PC's microprocessor how the microprocessor should communcate with the PC's memory and data busses. This code must be provided by the user to the microcontroller. When the C167CR is first powered on, it starts program execution from memory address 00'0000h. In order for the user's program to execute properly, a branch instruction to the start of the program must be located at memory address 00'0000h. 3.3.2

⁶In 'mainXYZ.asm' the XYZ is a unique alphanumeric identifier. For example, 'main114.asm' is the main file for the main assembly language file for CAN node 1 on the 14V bus.

⁷The locator calls the file 'linker.lnv'. This tells the locater where the Flash memory is located and where the RAM is located. This file is the same for all items on the CAN bus.

⁸The 'PATH' statement appears in both the 'Autoexec.bat' and Autoexec.dos' files in Windows95.

```
startupsec SECTION CODE ; codesection that contains reset pointer
sysreset PROC TASK INTNO=OH ; reset interrupt number is zero at Oh
ORG 000H ; forces next instruction to be located at Oh
JMP start ; installs a pointer to the startup routine
RETI ; return from interrupt
sysreset ENDP ; end procedure
startupsec ENDS ; end segment
```

Figure 3.8: C167CR Startup Code

The first instruction that is executed after the initial branch is typically 'DISWDT'. This istruction will disable the on chip watch dog timer. The watch dog timer is a timer that, if not serviced before a specific period of time, will reset the chip. This feature is not needed for the breadboard facility, so it is disabled.

After placing the appropriate branch instruction at memory address 00'000h and disabling the watch dog timer, the next thing that needs to be done is to tell the assembler and the linker about the memory that the C167CR can access. The CAN nodes for this network were made up of Phytec KitCON-167 boards. These kits are built around a CAN enabled Siemens C167CR microcontroller. They contained 256kbytes of on board flash memory, and 64kb of RAM. The memory map can be seen in Figure 3.9.

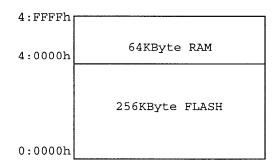




Figure 3.10: Assembly Code that allows External Memory Bus Accesses

meto:

```
NOP ; just loop here waiting
NOP
JMP meto
```

Figure 3.11: Loop Code for C167CR

The C167CR uses it's SYSCON, ADDRSEL and BUSCON registers to control access to off chip memory [6]. Figure 3.3.2 shows the code that would appropriately configure the microcontroller to access the memory on the KitCON-167 boards.

After the memory has been initialized, the 'EINIT' instruction has to be executed. This instruction locks in the memory configuration and allows further code to access the external memory. After this point, the SYSCON, ADDRSEL, and BUSCON registers cannot be changed. Once the 'EINIT' instruction has been executed, the system stack must be configured. After the stack is appropriately configured, each of the on chip peripherals that are to be used can now be configured. Configuration of an on chip peripheral is usually done by calling a function that is located in a different file. This is done as an organizational measure in keep file sizes small and readable. It also improves the abstraction layer between implementation of the software and the interface to that software. This allows the same 'main.asm' file to be used, with very little modification, for all sorts of different programs. Because configuration of most of the on chip peripherals is relatively simple, only the CAN bus initialization will be discussed in this thesis in Section 3.3.2.1.

Once all of the on chip peripherals have been initialized, the CPU must be set perform some sort of continuous loop. The code to do this is shown in Figure 3.3.2. Failure to cause the processor to loop will result in the processor to stop functioning at the end of the function.

3.3.2.1 The CAN Bus

Every CAN message contains 4 main user programmable parts. These parts are

- 1. Data Length Code
- 2. Message Direction
- 3. Arbitration Registers
- 4. Message Control Registers

Figure 3.12 shows how the major portions of a CAN message are arranged in memory. This grouping of registers in memory is referred to as a Message Object. The C167CR has 15 Message Objects. CAN is capable of transmitting variable length messages of up to 8 bytes in length. It is therefore, necessary to specify within the message, the length of the data field. This is done by setting the Data Length Code value in the Message Configuration Register. Next, each CAN message can either transmit data or receive data. Therefore it is necessary to specify this value by setting the Message Direction bit in the Message Configuration Register. Each CAN message has a unique message ID. This message ID is placed into the Upper Arbitration Register. Message IDs can either be 11 bits in length or they can be 29 bits long. For the purpose of this thesis, 11 bit message IDs have been used. Finally, every CAN message has a Message Control Register that specifies the behavior of the message object with respect to interrupts and how the message object will change when the data fields in the message object change.

3.3.3 Load Nodes

The load nodes were configured to be able to independently turn on and turn off multiple loads. Most nodes were configured to turn on and turn off 2 different loads, but some were configured to control as many as 3 loads. The nodes were also configured to collect current information provided by a node's smart switch's current sense pin. Each load node is able to report the current of each load and also the state (on or off) of each load when the appropriate command from the CAN bus is received. Table 3.3 and Table 3.4 show the messages⁹ for each load and the node that they

⁹These are not actually the CAN message numbers, but they are the contents of the Upper Arbitration Register of a CAN message Object from which the CAN message number is generated. In order to generate the actual CAN message ID, the first nibble in the Upper Arbitration Register would be moved into the lst position and then the entire word would be shifted to the right by one bit.

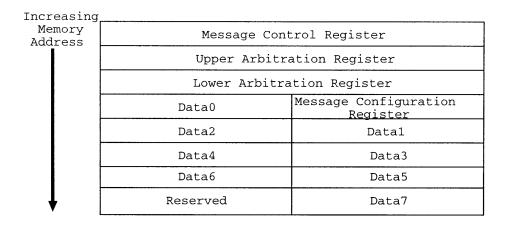


Figure 3.12: CAN Message Object Regsiters and Memory Locations

appear on. All messages marked Receive are configured to receive two different pieces of data. If the received datum is #000001h then the corresponding smart switch is turned on. If the received datum is #000800h then the corresponding smart switch is turned off.

3.3.4 Energy Management Node

The energy management node serves the purpose of both collecting the data necessary to make decisions involved with energy management, and to actually run the energy management algorithm itself. The algorithm was located on this node because it allowed easy access through memory to the collected data. It could, in fact, be located on any node on the network and the necessary data could be simply transmitted to that node across the network. The energy management algorithm is executed once every second. The last piece of data to be collected is the 42V current and direction information. After this datum is stored, the energy management algorithm function is called. The energy management algorithm produces an 8-bit pattern and sends this information across the network to the DC/DC converter node.

The energy management node is configured to collect voltage, current magnitude, current direction, and temperature for each of the batteries. The hardware necessary to collect battery temperature information was not implemented, so the software was written to collect, but ignore, the datum that the A/D collects when it is supposed to collect information about temperature. In total, this board has 6 A/D channels. Each channel is accessed once a second.

Breadboard Loads				
	14v Bus Node 1			
CAN Message	CAN Message Direction	CAN Message Number		
Power Door Locks	Receive	#0001h		
Seat & Door Module	Receive	#2001h		
Power Door Locks Current	Transmit	#6001h		
Seat & Door Current	Transmit	#4001h		
Power Door State	Transmit	#0010h		
Seat & Door State	Transmit	#0011h		
	14v Bus Node 2			
Turn Lights	Receive	#8001h		
Turn Lights Current	Transmit	#4007h		
Turn Lights State	Transmit	#0012h		
	14v Bus Node 3			
ABS	Receive	#C001h		
Brake Loads	Receive	#E001h		
ABS Current	Transmit	#E002h		
Brake Loads Current	Transmit	#0002h		
ABS State	Transmit	#0013h		
Brake State	Transmit	#0014h		
Bus Bridge	Receive	#0022h		
Bus Bridge Current	Transmit	#0023h		
Bus Bridge State	Transmit	#0024h		

Table 3.3: 14v Bus CAN Messages

The data is collected as the lower 10-bits of a word of memory. These 10-bits, however, represent a voltage from 0V to 5V not a current of up to 100 amps or a voltage of up to 60 volts. In order to properly use the information, it must be scaled. In the case of the voltage, it is not scaled on the microcontroller, instead, it is scaled and displayed in LabView. This is done because LabView takes care of much of the difficulty of using floating point numbers. In the case of the current, however, because the state of charge of each battery is calculated by integrating the total charge that has entered and exited each battery, it must be scaled on chip. The problem with scaling the measured number is that it could result in a loss of accuracy. This is undesirable, so instead of scaling the measured reading, the initial charge on each battery was scaled before assembling the code, and that scaled number is added to and subtrated from to compute the state of charge for each battery. The scaling for the initial state of charge for each battery was done as follows. First, the reserve capacity of the battery is multiplied by 15^{11} in order to compute the number of seconds that the battery can be discharged at 100 amps. Then, it must be realized that when the A/D converter produces the 10-bit pattern #03FFh it is actually reading 100 amps of current. If the current is measured every second, then the 10-bit pattern produced by the A/D converter is not only the current, but, by definition, it is also the total charge for one second. Multiplying #03FFh by the number of seconds that the battery can be discharged at 100 amps, returns the state of charge of the battery in a format that the output of the A/D can now be simply added and subtracted from with out any sort of conversion or loss of precision.

The initial value for the 36V battery was #01063E6h and the value for the 12V battery was #02576A0h. These two numbers are both larger than would be allowed by the 16-bit registers of the C167CR, so they are broken into two different words (a high word and a low word) and stored in two different variables in memory. The 10-bit output of the A/D converter is then added to the low word of the battery's state of charge, and, immediately afterward, zero and the carry bit is added to the upper word by using the add-carry instruction. These instructions are executed consecutively as atomic instructions so that they may not be interrupted inbetween and the carry bit be corrupted.

3.3.5 Serial to CAN Router Node

One of the goals of the breadboard facility was to try to explore possible useful functions of having an in-car automobile network. One possible benefit of the network would be in the area of self diagnostics. In the automobile of the future, because loads will be controlled by a digital network and connected off of a power bus, it will be much more difficult to tell where the fault in the network has occurred unless there were some catastrophic failure which left smoke, soot or other physical indicators that clearly indicate the culprit. In the absence of such physical indicators, it might be impossible to track down the fault unless the network has some intelligence and can tell the operator where the fault occurred. It is, therefore, necessary to be able to quickly and easily connect to the in-car network. If it were possible to interface to the in-car CAN network through a serial port, almost any device with a serial port¹² could be programmed to act as diagnostic

¹¹The multiple 15 is obtained because the reserve capacity of a battery is the number of minutes that a battery can be discharged at 25 amps. Multiply reserve capacity by 60 and the total number of seconds that the battery can be discharged at 25 amps is known. Divide this new number by 4 and the number of seconds that the battery can be discharged at 100 amps is known.

¹²Serial port in this case means an RS232 port

equipment for the automobile. Therefore, a serial to CAN router was written. This router employs time out error checking and checksum error checking.

In order to be able to translate between CAN and serial, it is necessary to develop rules that will convert a CAN message to a serial message. It is, therefore, necessary to understand the different parts of a CAN message that would come into play in such a translation. Section 3.3.2.1 discusses these parts in detail, but quickly below are the major user programmable parts.

- 1. Data Length Code
- 2. Message Direction
- 3. Arbitration Registers
- 4. Message Control Registers

The data necessary for each of these parts must be transmitted in the messages going from the PC to the Serial to CAN Router Node. They must then be moved into a CAN message object and transmitted onto the CAN bus. If the serial message sent is simply a command to turn something on the CAN bus on or off, the serial message is put into message object 1. If the message sent from the PC is a request for data, then message object 2 is used. The format of the serial message can be seen in Figure 3.13.

Figure 3.13: Format of Serial Message

All numbers and characters in Figure 3.13 are written in hexidecimal notation. Each character in the message represents a nibble¹³ of information. These bytes can be grouped into words or double words. Groups 1 and 14 represent the message delimiters. These are used to prevent LabView

 $^{^{13}\}mathrm{A}$ nibble is defined here as 4 bits.

from removing any leading edge zeros and thereby change the message length. These are not used in computing the checksum of the message. Group 2 represents the data length code. It has a data range of 0h to 8h. Group 3 represents the direction of transmission. It can have the value of either 8h for a transmit message or 0h for a receive message. Group 4 represents the the value that will be placed into the Upper Arbitration Register of the message object. From this value the actual message id of the CAN message can be obtained. Groups 5 through 12 represent the data bytes, but because of how the CAN router is written, only data in groups 6 and 7 will be transmitted, and they will be transmitted as one word with group 6 being the upper byte of the word. The value of #0800h in the 6/7 combination word indicates the the receiving node is to turn off a device, and the value #0001h in the 6/7 combination word indications that the receiving node is to turn on a device. Finally, group 13 represents the checksum of the message. The checksum is computed by simply adding up the values in groups 2, 3, 4, 6, and 7 on a byte by byte basis.

3.3.6 Data Collection Module

The data collection node was designed to prepare the batteries' voltages and currents so that the information could be converted from analog to digital and then used by the energy management algorithm. The information was converted from analog to digital via the Siemen's C167CR on chip 10-bit analog to digital converter. [7] The module was configured to measure voltage, current, and temperature for each battery; however, temperature was not used for this thesis. Because the A/D on the C167CR only has an input range of zero to five volts, all measured signals had to be preprocessed in get them within that range. The 36V battery voltage was measured by dividing the 36V battery's voltage by 11 and then reading that value. The 12V battery's voltage was measured by dividing its voltage by 5 and then reading that value. The current on each battery was measured by passing half the current for each bus through different hall effect current sensors. These sensors returned a current that was $\frac{1}{1000}$ times the sensed current. This current was sent through a 50 Ω resistor. This voltage, however, could be either positive or negative, so its absolute value was taken by the circuit in Figure 3.14. This circuit returned both the absolute value of the input, and it returned whether the current was into or out of the battery. If there was 5V on the "Current Direction" terminal, then the current was leaving the battery and if there was 0V on the "Current Direction" terminal then the current was entering the battery. A value of zero at the output of the current direction means that the battery is charging and a value of one at the output of the current direction means that the battery is discharging.

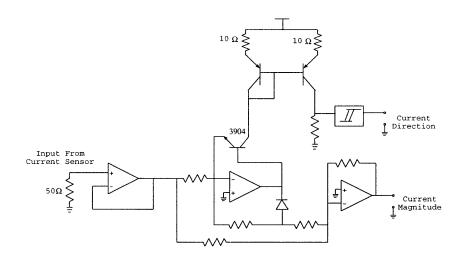


Figure 3.14: Precision Absolute Value Circuit with Direction SubCircuit

3.3.7 PC Input Files

One goal of the breadboard facility was to be able to allow tests that were run on Saber to be confirmed on the breadboard. The Saber simulations study "the effects of varying vehicle driving speeds and load events on power flow and energy usage [in order] to provide insite into the sizing of key power supply components such as the alternator, batteries, and DC/DC converter" [8]. In order to allow this, a program was written that would take in Saber formatted drive cycles and Saber formatted load cycles and convert them into a tab delimited format that could be read in by the breadboard facility. A copy of the first few lines of a breadboard input file can be seen in Figure 3.3.1.1. The program also takes in a list of the loads that are available on the breadboard facility and those loads' respective CAN Message ID's¹⁴

¹⁴CAN Message ID here refers to the value that is loaded into the Upper Arbitration Register of a CAN message object on a Siemens C167CR microcontroller. The actual Message ID can be derived from this value.

Breadboard Loads			
CAN Message	CAN Message Direction	CAN Message Number ¹⁰	
	42V Bus Node 1		
Brake by Wire	Receive	#0003h	
Heated Rear Windows	Receive	#4003h	
Brake by Wire Current	Transmit	#6003h	
Heated Rear Windows Current	Transmit	#2003h	
Brake by Wire State	Transmit	#0015h	
Heated Rear Window State	Transmit	#0016h	
	42v Bus Node 2		
Heater	Receive	#8003h	
Rear Seat Heater	Receive	#A003h	
Heater Current	Transmit	#C003h	
Rear Seat Heater Current	Transmit	#0019h	
Heater State	Transmit	#0017h	
Rear Seat Heater State	Transmit	#0018h	
	42v Bus Node 3		
Emissions Air Pump	Receive	#0004h	
Heated Windshield	Receive	#4004h	
Emissions Air Pump Current	Transmit	#2004h	
Heated Windshield Current	Transmit	#6004h	
Emmissions Air Pump State	Transmit	#0020h	
Heated Windshield State	Transmit	$\#001\mathrm{Ah}$	
DC	C/DC Converter Node		
DC/DC Converter Digital Input	Receive	$\#000\mathrm{Eh}$	
DC/DC Converter Input State	Transmit	#000Fh	
DC/DC Converter ON/OFF	Receive	#0021h	
	Data Collection Node		
42v Voltage	Transmit	#0005h	
42v Current & Direction	Transmit	#0006h	
42v Temperature	Transmit	#0007h	
42v State of Charge	Transmit	#0008h	
14v Voltage	Transmit	#0009h	
14v Current & Direction	Transmit	#00BAh	
14v Temperature	Transmit	#000Bh	
14v State of Charge	Transmit	#000Ch	

Table 3.4: 42v Bus CAN Messages

Chapter 3

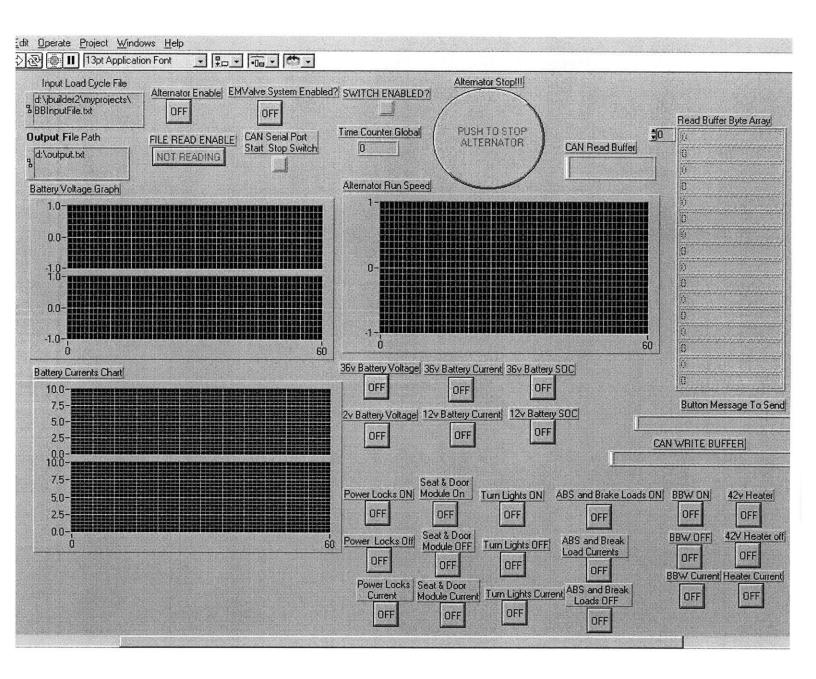


Figure 3.15: The LabView Breadboard Interface

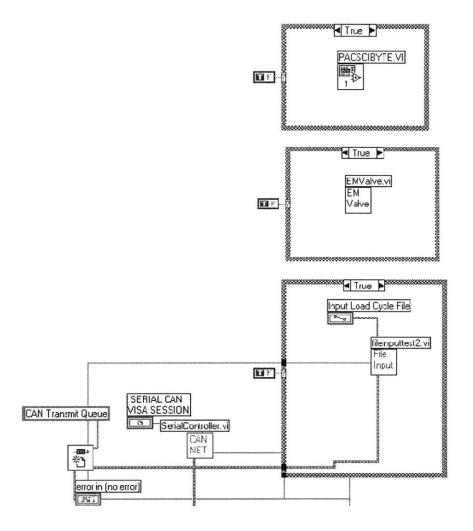


Figure 3.16: The major communicating subsystems

Chapter 4 Test Procedure

This chapter presents the test procedure which was used to measure the effectiveness of the battery voltage regulation energy management algorithm. Testing an energy management algorithm is a 6 stage process. These stages are listed below.

- 1. Design an energy management algorithm
- 2. Select a drivecycle to use with it
- 3. Design an appropriate electrical loadcycle for the selected drivecycle
- 4. Convert the drivecycle and loadcycle into a breadboard input file
- 5. Run the breadboard input file on the breadboard test facility
- 6. Analyze collected data

4.1 Design an Energy Management Algorithm

Energy management algorithm design and implementation is discussed in detail in Chapter 2 of this thesis.

4.1.1 Selecting a Drivecycles

A drivecycle is a data file which contains time, car velocity, and car gear in three columns. The drivecycle's information can be converted to alternator shaft speed using the Equation 4.1 [8], or engine shaft speed by using Equation 4.2.

Alternator Shaft Speed =
$$v * \frac{10}{36} * \frac{60}{\pi} * d * g_d * g_t * g_{e,a}$$
 (4.1)

Alternator Shaft Speed =
$$v * \frac{10}{36} * \frac{60}{\pi} * d * g_d * g_t$$
 (4.2)

The program that generates the breadboard input files actually calculates the engine shaft speed because it actaully controls the speed of the motor that drives the alternator, and that is connected to the alternator at a gearing of 3 to 1.

Variables	Variables Used in Car Velocity to Alternator Conversion			
Variable	Description	Ratio		
v	Vehicle Driving Speed [km/hr]			
d	Diameter of Vehicle's Tires [m]	0.594		
g_d	Differential Gear Ratio	4.0		
g_t	Transmission Gear Ratio			
	- Neutral	0		
	- 1^{st} Gear	3.071		
	- 2^{nd} Gear			
	- 3^{rd} Gear	1.194		
	- 4^{th} Gear	0.868		
	- 5^{th} Gear	0.700		
$g_{e,a}$	Engine-Alternator Gear Ratio	3.0		

Table 4.1: Variables Used in Car Velocity to Alternator Conversion

4.1.2 Loadcycles

An electrical loadcycle is a Saber *.scs input file that lists items by name, and lists those item's on and off times. The electrical loadcycle that was used with drivecycle "ece15.dat" was "winter worst ece15". The set of loads that was used for the test can be found in "breadboardloads.txt". Both "winter worst ece15" and "breadboardloads.txt" can be found in Appendix B.12

Drivecycle ece15.dat was selected because it has been tested and shown to work with SABER. As more drivecycles are proven to work with SABER, more will be used. It is the hope that algorithms can be tested on SABER and then verified using the breadboard system. Drivecycle ece15.dat will be matched with a slightly modified version of the electrical loadcycle "winter worst ece15". This electrical loadcycle was used by research unit number six and can be found at the end of this paper.

The goal of this test procedure is to allow the energy management algorithms to be tested on both a computer running Saber and on the MIT breadboard facility. Because the breadboard runs in real time, the hope is that the computer will help eliminate algorithms which don't make any sense and thus save time.

The tests will concentrate on the first two levels of sophistication. The third level will be investigated as part of future research. There will be two rounds of tests. The first series of tests will run using the 14-Volt Bus Regulation algorithm. This is the simplest algorithm and the easiest and cheapest to implement. The results of tests run using this algorithm will be used as a reference to measure the relative performance of the more sophisticated algorithms. The second series of tests will run using the Battery Model level algorithm. The results of these tests will be compared to the results from the 14-Volt Bus Regulation tests.

4.2 Test Procedure

1. Obtain LabView loadcycle.

This can be obtained by writing one from scratch or by translating a SABER drivecycle and loadcycle.

- 2. Determine number of times to run LabView loadcycle and enter value into LabView.
- 3. Power on breadboard facility.
- 4. Start Simulation.
- 5. Wait until all test runs have been completed.
- 6. Collect and analyze data.
- 7. Wait 24 hours and collect battery SOC data.

The data to be collected is

- Open circuit battery voltage before test
- Battery Voltages during test
- Open circuit Battery voltages after test

Chapter 5 Results and Conclusion

Tests were run and data was collected. The open circuit battery voltages before the tests were 36.51 volts and 13.82 volts. The final voltages for each battery (after 10 minutes of rest) were 36.19 volts and 13.22 volts. A plot of battery voltage against time during the test is shown below.

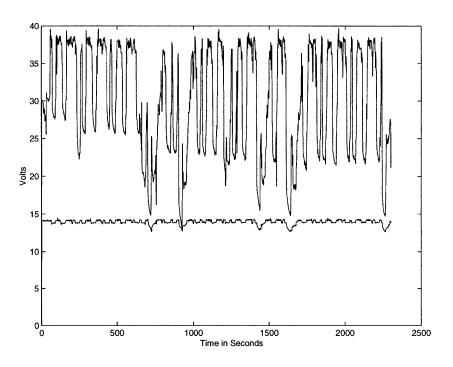


Figure 5.1: Battery Voltages vs Time

It is apparent from Figure 5.1 that the 36V battery's voltage vaired widely while the 12V battery was regulated to a very smooth voltage. This seems to indicate that the 36V battery was supplying

the 12V battery a considerable amount of power. This is one of the major flaws of the voltage regulation method of energy management. A more intelligent algorithm would be able to reduce the amount of current demanded by the DC/DC converter. That would have the effect of reducing the 14V bus, but it would also have the effect of reducing some of the ripple in the 42V bus. Although an advanced algorithm was designed and implemented for this thesis, there was not enough time to actually test it, so its results have not been included with the thesis.

The above data shows that the present system of simply regulating the voltage on each battery will probably no longer be adequate in the the 42V/14V dual voltage environment. It will, therefore, be helpful to further investigate energy management algorithms.

Appendix A

Complete Sophisticated Energy Management Algorithm

This algorithm was designed and implemented in software in the file ema.asm; however, because of time constraints, it was impossible to fully test it. The table can be read as follows. (12V SOC Region, 36V SOC Region). Negative battery current means that the batteries are draining.

SOC Region	12v Battery Current Sign	36v Battery Current Sign	DC/DC Converter Output
(1,1)	_	-	NONE
(1,1)	_	+	FULL
(1,1)	+	-	OFF
(1,1)	+	+	OFF
(1,2)	-	-	NONE
(1,2)	-	+	UP
(1,2)	÷	-	OFF
(1,2)	+	+	OFF
(1,3)	_	_	OFF
(1,3)	_	+	NONE
(1,3)	+	-	OFF
(1,3)	+	+	OFF
(1,4)	-	-	OFF
(1,4)	_	+	OFF
(1,4)	+	-	OFF
(1,4)	÷	+	OFF
(1,5)	-	-	OFF
(1,5)	-	+	OFF
(1,5)	+	_	OFF
(1,5)	+	+	OFF

Figure A.1: Decisions made when 12v Battery is in the "Dangerous Overcharge" Region

SOC Region	12v Battery Current Sign	36v Battery Current Sign	DC/DC Converter Output
(2,1)	-	-	UP
(2,1)	_	÷	FULL
(2,1)	+	-	NONE
(2,1)	+	+	FULL
(2,2)	-	-	NONE
(2,2)	-	÷	UP
(2,2)	÷	-	DOWN
(2,2)	+	+	OFF
(2,3)	-	-	NONE
(2,3)	-	+	NONE
(2,3)	+	-	DOWN
(2,3)	+	+	DOWN
(2,4)	-	-	OFF
(2,4)	_	+	OFF
(2,4)	+	-	OFF
(2,4)	+	+	OFF
(2,5)	-	-	OFF
(2,5)	-	+	OFF
(2,5)	+	-	OFF
(2,5)	+	+	OFF

Figure A.2: Decisions made when 12v Battery is in the "Acceptable Overcharge" Region

SOC Region	12v Battery Current Sign	36v Battery Current Sign	DC/DC Converter Output
(3,1)	-	_	FULL
(3,1)	-	+	FULL
(3,1)	+	-	FULL
(3,1)	+	+	FULL
(3,2)	-	-	FULL
(3,2)	-	+	FULL
(3,2)	+	-	FULL
(3,2)	+	+	FULL
(3,3)	-	-	NONE
(3,3)	-	+	NONE
(3,3)	+	-	NONE
(3,3)	+	+	NONE
(3,4)	-	_	DOWN
(3,4)	_	+	DOWN
(3,4)	+	-	DOWN
(3,4)	+	+	DOWN
(3,5)	-	-	OFF
(3,5)	-	+	OFF
(3,5)	+	_	OFF
(3,5)	+	+	OFF

Figure A.3: Decisions made when 12v Battery is in the "Ideal Operation" Region

SOC Region	12v Battery Current Sign	36v Battery Current Sign	DC/DC Converter Output
(4,1)	-	-	FULL
(4,1)	-	+	FULL
(4,1)	+	-	FULL
(4,1)	+	+	FULL
(4,2)	-	-	FULL
(4,2)	-	+	FULL
(4,2)	+	-	FULL
(4,2)	+	+	FULL
(4,3)	_	-	UP
(4,3)	_	+	UP
(4,3)	+	-	UP
(4,3)	+	+	UP
(4,4)	-	-	DOWN
(4,4)	-	+	UP
(4,4)	+	-	DOWN
(4,4)	+	+	NONE
(4,5)	-	-	OFF
(4,5)	-	+	UP
(4,5)	+	-	OFF
(4,5)	+	+	OFF

Figure A.4: Decisions made when 12v Battery is in the "Acceptable Undercharge" Region

SOC Region	12v Battery Current Sign	36v Battery Current Sign	DC/DC Converter Output
(5,1)	-	-	UP
(5,1)	_	+	UP
(5,1)	+	_	UP
(5,1)	+	+	UP
(5,2)	-	_	UP
(5,2)	-	+	UP
(5,2)	+	-	UP
(5,2)	+	+	UP
(5,3)	_	_	UP
(5,3)	-	+	UP
(5,3)	+	_	UP
(5,3)	+	+	UP
(5,4)	_	_	DOWN
(5,4)	-	+	NONE
(5,4)	+	_	DOWN
(5,4)	+	+	NONE
(5,5)	-	_	OFF
(5,5)	-	+	OFF
(5,5)	+	-	OFF
(5,5)	+	+	OFF

Figure A.5: Decisions made when 12v Battery is in the "Dire Undercharge" Region

Appendix B Breadboard Code

B.1 Organization

This appendix contains the complete code for all items used in the bread board facility.

- 1. 14V Bus CAN Node 1 B.2
- 2. 14V Bus CAN Node 2 $\mathrm{B.3}$
- 3. 14V Bus CAN Node 3 B.4 $\,$
- 4. 42V Bus CAN Node 1 B.5
- 5. 42V Bus CAN Node 2 $\mathrm{B.6}$
- 6. 42V Bus CAN Node 3 B.7
- 7. CAN Router B.8
- 8. Data Acquisition Node B.8
- 9. DC/DC Converter Node B.10
- 10. Saber to Breadboard Converter Code B.11
- 11. Breadboard Loads B.12

B.2 14V Bus CAN Node 1

On the next page starts the code for the 14V bus CAN node 1. The files for the node are as follows.

1. comp112.bat

- 2. main112.asm
- 3. cnmod112.asm
- $4. \ canmol112.asm$
- 5. cnint112.asm
- 6. atod112.asm
- 7. tmrs112.asm
- 8. linker.lnv
- $9. \ {\rm Reg167b.def}$



comp112.bat

al66 main112.asm al66 cnmodl12.asm al66 canmol12.asm al66 caint112.asm al66 atod112.asm al66 tmrs112.asm al66 tmrs112.asm l166 LINK main112.obj cnmodl12.obj canmol12.obj cnint112.obj atod112.obj tmrs112.obj TO locatein.lno l166 @linker.lnv ihex166 -il6 locate.out -o main112.hex



main112.asm



:: Initialize CAN Bus SSEGMENTED ; Call the CAN initialization function CALL canin SEXTEND ;; End of CAN Bus Initialization SEXTSFR ; CAN USE ALL internal RAM for Stack \$EXTSSK meto: SEXTMEM ; just loop here waiting NOP \$NOMOD166 \$STDNAMES(reg167b.def) NOP JMP meto SSYMBOLS RET ; return main ENDP NAME main mainseg ENDS RBANK1 COMREG R0-R15 ; define a common register area of 16 register startupsec SECTION CODE ; codesegment that contains reset int pointer ; default stack size of 256 Words SSKDEF 4 sysreset PROC TASK INTNO=0H ; reset interrupt number is zero at Oh ORG 000H ; forces next instruction to be located at Oh ASSUME DPP3:SYSTEM JMP start ; installs a pointer to the startup routine ; return from interrupt RETT EXTERN canin: FAR ; Can function sysreset ENDP ; external atod initialization EXTERN atod initialize:FAR startupsec ENDS EXTERN atod timer initialize:FAR END mainseg SECTION CODE main PROC FAR start: DISWDT ; disable the watchdog timer BSET IEN ; Globally Enable Interrupts both global ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh EINIT ; end initialization ;; End of external memory bus initialization ;; Initialize the Data Page pointers for this section MOV DPP3, #03h ; make DPP3 point to system ;; End of Data Page Pointer Initialization ;; Make the direction of Port 2 to output MOV DP2, ONES :: Make sure Port 2 is in push/pull mode MOV ODP2, ONES ;; Initialize The Stack ;; The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #0FBFFh the highest ;; byte that the stack pointers can point to is #OFBFEh MOV STKUN, #0FBFEh; Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP, #0FBFEh ; Set the Stack Pointer :: End of Stack Initialization ;; Initialize the Analog to Digital Converter CALL atod initialize; atod :: End of A/D initialization ;; Initialize A/D timer CALL atod_timer_initialize; timers ;; End of A/D timer initialization



cnmod112.asm



		enniou	12.45111	
SEGMEN	IMED		RET	
\$EXTEND \$EXTSFR)		canin ENDP	
\$EXTSFR \$EXTMEM			setall PROC FAR	; This Procedure sets all of the Mess objs invalid
\$NOMOD1			;; by using a counter	it counts up to 15 and initializes all of the message
	MES(reg167b.def)		;; objects along the	
\$SYMBOL			PUSH R2	
			PUSH R4	
NAME ca	anmod		PUSH R5	
NUCLARRY DURING			AND R5, ZEROS	Orthogonal to 1 from filmet NO
	COMREG R0-R15	; define a common register area of 16 registers	OR R5, #01h	; Set counter to 1 for first MO
GLOBAL	canin	; The function must be declared Global at the	AND R2, ZEROS	Cat paintage to MOI
		; beginning of the module	OR R2, #0EF10h	; Set pointer to MO1
	6- DDD	andimuna anggifig Magazga phiogta	AND R4, ZEROS OR R4, #5555h	; Set R4 to make MObs invalid
EXTERN	canmocfg:FAR	; configures specific Message objects	OK K4, #555511	, Set N4 to make Mobs invalle
ACCIME	DPP3:SYSTEM		nextreg:MOV [R2],R4	; make all message objects invalid
ASSUME	DFF5:5151EM		ADD R2, #10h	,
canfunc	SECTION CODE	; codesegment that contains reset int pointer	CMPI1 R5, #0Fh	
cumuno	berron copp	,	JMPA CC_NZ, nextreg	;
canin	PROC FAR		POP R5	
	PUSH RO		POP R4	
	PUSH R1		POP R2	
			RET	
		e CAN control registers	setall ENDP	
		; set control register to zero	canfunc ENDS	
	MOV R1, #0043h	; Set IE and INIT bits	END	
	OR C1CSR, R1 ;	set control register to R1's value	END	
	AND CIDER 7EDOC	; set Bit timing register to zero		
	MOV R1, #03447h	; set for 125k operation		
		; set Bit timing register parameters		
	on cibin, ni	, bee bis climing rogicour persinentia		
	AND C1GMS, ZEROS	; set Global Mask short register to zero		
	MOV R1, #0FFFFh	; EOFF is what DAVE initialize		
	OR CIGMS, R1	; set GMS		
		S ; set Upper global mask long to zero		
	MOV R1, #0FFFFh			
	OR CIUGML, R1			
	MOU D1 #0E9EEb			
	MOV R1, #0F8FFh AND C1LGML, ZEROS	S		
	OR CILGML, R1	; lower global mask		
	on cibonb, ni	, ionol global matri		
	AND C1UMLM, ZEROS	S		
	OR CIUMLM, R1	; upper mask of last register		
	AND C1LMLM, ZEROS			
	OR C1LMLM, R1	; lower mask of last register		
	CALL setall	; sets all of the CAN registers to off		
	CALL canmocfg	; Configures specific Message Objects		
	·· Setup CAN inte	errupt and Initialize CAN module		
E	EXTR #4			
	AND XPOIC, ZEROS	; configure CAN interrupt control Register		
	AND R0, ZEROS			
	OR R0,#0073h	; enable interrupt, level is 10 group is 2		
	방법이 잘 잘 걸었다. 여기 봐야 아름다. 아이는 것 같아요.	; Configure CAN interrupt Control Register		
	AND R1, ZEROS	Le le ville dout seconde the DOD		
		; crashes if I clear the CPU access to the BTR		
	sentences. The literature of the second second	; end initialize CAN interrupt		
	POP R1 POP R0			
	FOP RU			
			1	

99/05/09 12:44:38	canmo1	12.asm		1
<pre>\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS NAME canmo RBANK1 COMREG R0-R15 GLOBAL canmocfg</pre>	; declare bank of 16 global registers	f data	MOV [R2],R1; set M03's Control registerADD R2,#2h; point to Upper Arbitration registAND R3, ZEROS; set R6 to zeroOR R3, #06001h; The number is the Message ID for 1MOV [R2],R3; message id = 0ADD R2, #2h; Point to the Lower Arbitration ReMOV [R2], ZEROS; standard Message object so loweraAND R1, ZEROS; put 000h into first data byte andMOV MCD_M3,R1; Databyte(0) = 0 and Set to receamov DATA_M3, ZEROS; Fill the Data of the M0 with Zero	Message Object 3 egister arb = Oh a set to receive eive and 3 bytes o
;; the comment "Setup C ;; nothing to prevent t MOV R2, #MCR_M1 AND R1, ZEROS OR R1, #5599h tivates		f data	<pre>;; Initialize Message Object 4 MOV R2, #MCR_M4 ; start of Message Object 4 AND R1, ZEROS OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register ADD R2,#2h ; point to Upper Arbitration regist AND R3, ZEROS ; set R6 to zero OR R3, #04001h ; The number is the Message ID for MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Re MOV [R2], ZEROS ; standard Message object so lowera AND R1, ZEROS OR R1, #0038h ; put 0AAh into first data byte and MOV MCD_M4,R1 ; Databyte(0) = 0 and Set to rece a MOV DATA_M4, ZEROS ; fill the data of the M0 with ZERO</pre>	Message Object 4 egister arb = Oh d set to receive eive and 3 bytes o
ADD R2,#2h AND R3, ZEROS	; start of Message Object 2 ; RECEIVE INTERRUPT enabled [O2's Control register ; point to Upper Arbitration register ; set R6 to zero	f data	MOV DATA_M5, ZEROS ; fill the data of the MO with ZERO	Message Object 5 egister arb = Oh d set to receive eive and 3 bytes o
AND RS, ZEROS OR R3, #02001h MOV [R2],R3 ADD R2, #2h MOV [R2], ZEROS AND R1, ZEROS OR R1, #0030h MOV MCD_M2,R1 ta MOV DATA_M2, ZEROS ;; Initialize Message (MOV R2, #MCR_M3 AND R1, ZEROS OR R1, #5595h tivates	<pre>; The number is the Message ID for Message Object 2 ; message id = 0 ; Point to the Lower Arbitration Register ; standard Message object so lowerarb = 0h ; put 000h into first data byte and set to receive ; Databyte(0) = 0 and Set to receive and 3 bytes of da ; Fill the Data of the MO with Zeros</pre>	f data	<pre>;; Initialize Message Object 6 MOV R2, #MCR_M6 ; start of Message Object 6 AND R1, ZEROS OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register ADD R2,#2h ; point to Upper Arbitration regist AND R3, ZEROS ; set R6 to zero OR R3, #00011h ; The number is the Message ID for MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Re MOV [R2], ZEROS ; standard Message object so lowera AND R1, ZEROS OR R1, #0038h ; put 0AAh into first data byte and MOV MCD_M6,R1 ; Databyte(0) = 0 and Set to rece</pre>	Message Object 6 egister arb = Oh d set to receive



canmo112.asm



MOV DATA_M6, ZEROS

; fill the data of the MO with ZEROS

POP R3 POP R2 POP R1 RET canmocfg ENDP

can_module ENDS END

99/05/09 11:40:10

\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS

NAME canint RBANK1 COMREG R0-R15

; declare bank of 16 global registers

ASSUME DPP3:SYSTEM

can_interrupts SECTION CODE

can_receive_interrupt PROC TASK INTNO=040h ORG 0100h CALL can_receive_interrupt_handler RETI can_receive_interrupt ENDP

can_receive_interrupt_handler PROC FAR PUSH R0

PUSH R1 PUSH R2 MOVB RL0, INTID ; Read the CAN interrupt ID buffer CMPB RL0, #03h ; See if the interrupt came from M01 JMP cc_Z, message_one_interrupt; if interrupt from M01 handle MOV R1, #05555h MOV R2, #05599h MOV MCR_M2, R1 MOV R0, DATA_M2 MOV MCR_M2, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch

MOV R2,MCR_M6 MOV MCR_M6, R1 MOV DATA_M6, R0 MOV MCR_M6, R2 CMP R0, #01h JMP cc_NZ, turn_off_heated_rear_window BSET P2.1 JMP exit_function

JMP cc_NZ, exit_function BCLR P2.1 JMP exit_function

message_one_interrupt:

MOV R1, #05555h MOV R2, #05599h MOV MCR_M1, R1 MOV R0, DATA_M1 MOV MCR_M1, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch

MOV R2, MCR_M5 MOV MCR_M5, R1

cnint112.asm

MOV DATA_M5, R0

MOV MCR_M5, R2 CMP R0, #01h JMP cc_NZ, turn_heater_off BSET P2.0 JMP exit_function

turn_heater_off: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.0 exit_function: MOV R2, #0EFFFh AND C1CSR, R2 POP R2 POP R1 POP R0 RET

can_receive_interrupt_handler ENDP

can_interrupts ENDS END



atod112.asm



PUSH R3 SSEGMENTED PUSH R4 SEXTEND PUSH MDH \$EXTSFR PUSH MDL : CAN USE ALL internal RAM for Stack SEXTSSK \$EXTMEM MOV R2, ADDAT \$NOMOD166 : This is so we can isolate the A/D channel from whi MOV RO. R2 \$STDNAMES(reg167b.def) ch the data is coming SSYMBOLS : This is so we can isolate the A/D data and then sc MOV R3, R2 ale it by name atod ;; This code scales the data from the A/D by 21 to get the actual current fl owing through the BTS550P ASSUME DPP3:SYSTEM AND R3, #003FFh ; This isolates the lower ten bits of the A/D's output RBANK1 COMREG R0-R15 MOV R4, #01h ; There is no scaling done on the controller GLOBAL atod_initialize AND R0, #0F000h ; The channel information is located in the upper nibble ;; This A/D is set up to measure the current in two different CMP R0, #01000h ; See if the information is coming from Channel 1 of the A/ ;; loads. Because this software is to be used as part of D ;; 42volt bus node 1, it uses the names of the loads that JMP cc_Z, Rear_Seat_Heater_current ;; that node is supposed to control. ;; The analog to digital converter uses Port 5 MOV R0, #05555h ; This bit pattern deactives MCRs MOV R1, MCR_M3 ; SAVE the Configuration of the MCR atod setup SECTION CODE ; Kill the Message Control Register MOV MCR_M3, R0 atod_initialize PROC FAR ;; This gets the actual current value ;; Initialize variables MUL R3, R4 ; The output goes entirely into MDL NOP ;; This below line of code setups up the A/D converter ; Move the actual current value from the MDL registe MOV DATA M3, MDL ;; for 2 channels and single conversion. ;; It is also set for "Wait for read mode" r into the CAN message object MOV MCR_M3, R1 ;; so the converter will wait for the user program to read BSET T3R ;; the buffer before processing the next channel. ; setup A/D control register JMP exit_routine MOV ADCON, #0A221h ;; Set the channel to which the data should be written Rear Seat Heater_current: ;; when the first "A/D is done" interrupt occurs MOV R0, #05555h ; This bit pattern deactives MCRs : SAVE the Configuration of the MCR ;; The below code sets up the A/D's Interrupt control register MOV R1, MCR_M4 MOV MCR_M4, R0 ; Kill the Message Control Register :: The A/D is setup to have a group of 2 and a level of 10 :: This code tells me when I have completed a conversion on both channels MOV ADCIC, #006Fh ;; If the leds on port 2 are not counting then You know that the system isn' RET t performing conversionsS atod initialize ENDP MOV R0, #04h ;test code atod setup ENDS ADD P2, RO ;test code atod handlers SECTION CODE :: This generates the acutal current value atod_handler PROC TASK INTNO=028h ; The output goes entirely into MDL MUL R3, R4 ORG 0A0H NOP CALL atod_function MOV DATA_M4, MDL ; for testing purposes RETI MOV MCR M4, R1 atod handler ENDP exit_routine: atod function PROC FAR POP MDL ;; this function works by seeing if the converter is converting ;; for the heater_measurement. If the bit is set, then POP MDH ;; the bit gets cleared and the IP jumps to where the POP R4 POP R3 ;; value in the converter is moved into the heater_current POP R2 ;; variable. POP R1 :: otherwise the bit gets set and the value is moved into POP RO ;; the heated_rear_window_current variable RET PUSH R0 atod function ENDP PUSH R1 atod handlers ENDS PUSH R2



atod112.asm



END



tmrs112.asm

\$SEGMENTED ; These are assembler controls \$EXTEND SEXTSFR \$EXTMEM \$EXTINSTR \$NOMOD166 \$STDNAMES(reg167b.def) ; Assembler controls end here \$SYMBOLS NAME timer_functions ASSUME DPP3:SYSTEM RBANK1 COMREG R0-R15 GLOBAL atod_timer_initialize atod_timer SECTION CODE atod_timer_initialize PROC FAR ; setup Core Timer T3 MOV T3CON, #0004h MOV T3IC, #002Bh ; Make the value in the counter equal to zero MOV T3, #0000h ; enable the timer interrupt BSET T3IE ; start the timer BSET T3R RET atod_timer_initialize ENDP atod_interrupt PROC TASK INTNO=023h ORG 08Ch CALL atod_timer_handler RETI atod_interrupt ENDP atod_timer_handler PROC FAR BCLR T3R ; stop the timer ; start an A/D conversion BSET ADST RET atod_timer_handler ENDP atod_timer ENDS END



LOCATE locatein.lno (GENERAL) IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (04000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO locate.out





99/05/08 23:03:14

reg167b.def

A Discount of the other				

;** @(#)reg167k		1.10 12/18/97		
;**				
	efiniti	ons for the SAB C167		
		s all SFR names and BIT names		
;** This file o	can be	supplied to rm166 and a166 (STDNAMES control)		
; * * * * * * * * * * * * * * *	*****	**************************************		
TRUE	DEFB	OFF20h.0, RW		
NODE142	DEFB	OFF20h.1, RW		
CICSR	DEFA	OEFOOh		
INTID	DEFA	0EF02h 0EF04h		
C1BTR	DEFA DEFA	0EF06h		
C1GMS C1UGML	DEFA	0EF08h		
CILGML	DEFA	OEFOAh		
CIUMLM	DEFA	OEFOCh		
CILMLM	DEFA	OEFOEh		
MCR_M1	DEFA	0EF10h		
MCR_M2	DEFA	0EF20h		
MCR_M3	DEFA	0EF30h		
MCR_M4	DEFA	0EF40h		
MCR_M5	DEFA	0EF50h		
MCR_M6	DEFA	OEF60h		
MCR_M7	DEFA	0EF70h		
MCR_M8	DEFA	0EF80h		
MCR_M9	DEFA DEFA	0EF90h 0EFA0h		
MCR_MA MCR_MB	DEFA	0EFB0h		
MCR_MC	DEFA	OEFCOh		
MCR_MD	DEFA	0EFD0h		
MCR_ME	DEFA	OEFEOh		
MCR_MF	DEFA	0EFF0h		
MCD_M1	DEFA	0EF16h		
MCD_M2	DEFA	0EF26h		
MCD_M3	DEFA	0EF36h		
MCD_M4	DEFA	0EF46h		
MCD_M5	DEFA	0EF56h		
MCD_M6	DEFA	0EF66h		
MCD_M7	DEFA	0EF76h		
MCD_M8	DEFA	0EF86h 0EF96h		
MCD_M9 MCD_MA	DEFA DEFA	0EFA6h		
MCD_MB	DEFA	0EFB6h		
MCD_MC	DEFA	0EFC6h		
MCD_MD	DEFA	0EFD6h		
MCD_ME	DEFA	0EFE6h		
DATA_M1	DEFA	0EF18h		
DATA_M2	DEFA	0EF28h		
DATA_M3	DEFA	0EF38h		
DATA_M4	DEFA	0EF48h		
DATA_M5	DEFA	0EF58h		
DATA_M6	DEFA	0EF68h		
DATA_M7	DEFA	0EF78h		
DATA_M8	DEFA	0EF88h		
DATA_M9	DEFA	0EF98h 0EFA8h		
DATA_MA DATA_MB	DEFA DEFA	0EFB8h		
DATA_MB DATA_MC	DEFA	0EFC8h		
DATA_MD	DEFA	0EFD8h		
DATA_ME	DEFA	0EFE8h		
2004-000-0				
DP8	DEFR	OFFD6h		

P8	DEFR	0FFD4h
DP7	DEFR	0FFD2h
P7	DEFR	0FFD0h
DP6	DEFR	OFFCEh
P6	DEFR	OFFCCh
DP4	DEFR	OFFCAh
P4	DEFR	0FFC8h
DP3	DEFR	0FFC6h
P3	DEFR	0FFC4h
DP2	DEFR	0FFC2h
P2	DEFR	0FFC0h
SSCCON	DEFR	0FFB2h
SOCON	DEFR	OFFBOh
WDTCON	DEFR	OFFAEh
TFR	DEFR	0FFACh
P5	DEFR	0FFA2h
ADCON	DEFR	0FFA0h
T1IC	DEFR	0FF9Eh
TOIC	DEFR	0FF9Ch
ADEIC	DEFR	0FF9Ah
ADCIC	DEFR	0FF98h
CC15IC	DEFR	0FF96h
CC14IC	DEFR	0FF94h
CC13IC	DEFR	0FF92h
CC12IC	DEFR	0FF90h
CC11IC	DEFR	0FF8Eh
CC10IC	DEFR	0FF8Ch
CC9IC	DEFR	0FF8Ah
CC8IC	DEFR	0FF88h
CC7IC	DEFR	0FF86h
CC6IC	DEFR	0FF84h
CC5IC	DEFR	0FF82h
CC4IC	DEFR	0FF80h
CC3IC	DEFR	0FF7Eh
CC2IC	DEFR	0FF7Ch
CC1IC	DEFR	0FF7Ah
CCOIC	DEFR	0FF78h
SSCEIC	DEFR	0FF76h
SSCRIC	DEFR	0FF74h
SSCTIC	DEFR	0FF72h
SOEIC	DEFR	0FF70h
SORIC	DEFR	0FF6Eh
SOTIC	DEFR	0FF6Ch
CRIC	DEFR	0FF6Ah
TGIC	DEFR	0FF68h
T5IC	DEFR	0FF66h
T4IC	DEFR	0FF64h
T3IC	DEFR	0FF62h
T2IC	DEFR	0FF60h
CCM3	DEFR	0FF58h
CCM2	DEFR	0FF56h
CCM1	DEFR	0FF54h
CCM0	DEFR	0FF52h
T01CON	DEFR	0FF50h
T6CON	DEFR	0FF48h
T5CON	DEFR	0FF46h
T4CON	DEFR	0FF44h
T3CON	DEFR	0FF42h
T2CON	DEFR	0FF40h
PWMCON1	DEFR	0FF32h
PWMCON0	DEFR	0FF30h
CCM7	DEFR	0FF28h
CCM6	DEFR	0FF26h
CCM5	DEFR	0FF24h
CCM4	DEFR	0FF22h

99/05/08 23:03:14		
T78CON	DEFR	0FF20h
P1H	DEFR	0FF06h
P1L	DEFR	0FF04h
POH	DEFR	0FF02h
POL	DEFR	OFFOOh
PECC7	DEFR	OFECEh
PECC6	DEFR	OFECCh
PECC5	DEFR	0FECAh 0FEC8h
PECC4 PECC3	DEFR	0FEC6h
PECC2	DEFR	0FEC4h
PECC1	DEFR	0FEC2h
PECCO	DEFR	OFECOh
SRCP0	DEFA	OFCEOh
DSTP0	DEFA	0FCE2h
SRCP1	DEFA	0FCE4h
DSTP1	DEFA	0FCE6h
SRCP2	DEFA	0FCE8h
DSTP2	DEFA DEFA	0FCEAh 0FCECh
SRCP3 DSTP3	DEFA	OFCEEh
SRCP4	DEFA	OFCFOh
DSTP4	DEFA	0FCF2h
SRCP5	DEFA	0FCF4h
DSTP5	DEFA	0FCF6h
SRCP6	DEFA	0FCF8h
DSTP6	DEFA	OFCFAh
SRCP7	DEFA	0FCFCh
DSTP7	DEFA	OFCFEh
SOBG	DEFR	OFEB4h OFEB2h, r
SORBUF	DEFR DEFR	OFEB2h, r OFEB0h, w
SOTBUF WDT	DEFR	OFEAEh, r
ADDAT	DEFR	0FEA0h
CC15	DEFR	0FE9Eh
CC14	DEFR	0FE9Ch
CC13	DEFR	0FE9Ah
CC12	DEFR	0FE98h
CC11	DEFR	0FE96h
CC10	DEFR	0FE94h
CC9	DEFR	0FE92h 0FE90h
CC8 CC7	DEFR DEFR	0FE8Eh
CC6	DEFR	0FE8Ch
CC5	DEFR	0FE8Ah
CC4	DEFR	0FE88h
CC3	DEFR	0FE86h
CC2	DEFR	0FE84h
CC1	DEFR	0FE82h
CCO	DEFR	0FE80h
CC31	DEFR	0FE7Eh
CC30	DEFR	0FE7Ch
CC29 CC28	DEFR DEFR	0FE7Ah 0FE78h
CC28	DEFR	0FE76h
CC26	DEFR	0FE74h
CC25	DEFR	0FE72h
CC24	DEFR	0FE70h
CC23	DEFR	0FE6Eh
CC22	DEFR	0FE6Ch
CC21	DEFR	0FE6Ah
CC20	DEFR	0FE68h
CC19	DEFR	0FE66h 0FE64h
CC18	DEFR DEFR	0FE62h
CC17	DEFR	01 20211

reg167b.def

CC16			DEFR		OFE60	h
TIRE			DEFR		OFE56	
TORE			DEFR		OFE54	
TI	,ш		DEFR		OFE52	
TO			DEFR		OFE50	
	-					
CAPR	EL		DEFR		OFE4A	
т6			DEFR		OFE48	
т5			DEFR		OFE46	
т4			DEFR		OFE44	
т3			DEFR		OFE42	
т2			DEFR		OFE40	h
PW3			DEFR		OFE36	h
PW2			DEFR		0FE34	h
PW1			DEFR		OFE32	h
PW0			DEFR		OFE30	h
; Ex	tended	sfr	area			
,						
ODP8	1		DEFR		0F1D6	h
ODP7			DEFR		OF1D2	
ODP			DEFR		OF1CE	
					OFICE	
ODP3			DEFR			
PICC			DEFR		0F1C4	
ODP2			DEFR		0F1C2	
EXIC			DEFR		0F1C0	
SOTE			DEFR		0F19C	
XP31	C		DEFR		0F19E	h
XP21	C		DEFR		0F196	h
XP11	C		DEFR		0F18E	h
XP01	C		DEFR		0F186	h
PWMI	C		DEFR		0F17E	h
TSIC	2		DEFR		0F17C	h
T7I0			DEFR		0F17A	
CC31			DEFR		0F194	
CC30			DEFR		0F18C	
			DEFR		0F184	
CC29					0F178	
CC28			DEFR			
CC27			DEFR		0F176	
CC26			DEFR		0F174	
CC25			DEFR		0F172	
CC24	IC		DEFR		0F170	
CC23	BIC		DEFR	1	0F16E	
CC22			DEFR		0F16C	h
CC21	LIC		DEFR		0F16A	h
CC20	DIC		DEFR		0F168	h
CC19	DIC		DEFR		0F166	h
CC18	BIC		DEFR		0F164	h
CC17	7IC		DEFR		0F162	h
CC16			DEFR		0F160	h
RPOH	ł		DEFR		0F108	h
DP1H			DEFR		0F106	
DP11 DP11			DEFR		0F104	
100 To 100			DEFR		0F104	
DPOH					0F102	
DP01			DEFR			
SSCI			DEFR		OFOB4	
SSCI			DEFR		OFOB2	
SSC			DEFR		OFOBO	
ADDA			DEFR		OFOAC	
T8R	EL		DEFR	9	0F056	h
T7RI	EL		DEFR		0F054	h
т8			DEFR		0F052	h
т7			DEFR	6	0F050	h
PP3			DEFR		OF03E	h
PP2			DEFR		0F030	
PP1			DEFR		OF03A	
LLT.			DLI N	5	0. 00F	



99/05/08	9-e
23:03:14	Scott
20.00.14	

DEED	0503.85	AN13	DEFB	P5.13
DEFR	0F036h			P5.14
DEFR	0F034h	AN15	DEFB	P5.15
		TGEUD	LTT	'AN10'
				'AN11'
DEFR	0F030h			
		TGIN	LIT	'AN12'
		T5IN	LIT	'AN13'
-				'AN14'
DEFB	P2.1	T2EUD	LIT	'AN15'
DEEB	P2 2			
		POLITIO	DEEB	P7.0
DEFB	P2.4			P7.1
DEEB	P2 5	POUT2	DEFB	P7.2
		POLITIS	DEEB	P7.3
				P7.4
DEFB	P2.7			
DEFB	P2 . 8	CC29I0	DEFB	P7.5
		CC30I0	DEFB	P7.6
				P7.7
DEFB		003110	DEFD	£7.7
DEFB	P2.11			
		CC16I0	DEFB	P8.0
			DEEB	P8.1
DEFB	P2.14			P8.2
DEFB	P2.15	CC19I0	DEFB	P8.3
		CC20T0	DEFB	P8.4
LIT	'CC110'			P8.5
LTT	'CC210'	CC22I0	DEFB	P8.6
		CC23TO	DEFB	P8.7
PT.L.	66310	002010		
DEFB	P3.0			
		TOM	DEFB	T01CON.3
				TO1CON.6
DEFB	P3.3	TIM	DEFB	T01CON.11
	P3 4	T1R	DEFB	T01CON.14
	P3.7	т7М	DEFB	T78CON.3
DEFB				
DEFB	P3.6	T7R	DEFB	T78CON.6
DEFB	P3.6			
DEFB DEFB	P3.6 P3.5	T7R T8M	DEFB DEFB	T78CON.6 T78CON.11
DEFB DEFB DEFB	P3.6 P3.5 P3.8	T7R	DEFB	T78CON.6
DEFB DEFB	P3.6 P3.5	T7R T8M T8R	DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14
DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9	T7R T8M	DEFB DEFB	T78CON.6 T78CON.11
DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10	T7R T8M T8R ACC0	DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3
DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11	T7R T8M T8R ACC0 ACC1	DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13	T7R T8M T8R ACC0 ACC1 ACC2	DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11
DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11	T7R T8M T8R ACC0 ACC1	DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13	T7R T8M T8R ACC0 ACC1 ACC2	DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15	T7R T8M T8R ACC0 ACC1 ACC2 ACC3	DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15
DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.5 P4.6	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC9 ACC10	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.5 P4.6	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC9 ACC10	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC9 ACC10	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P4.0 P4.1 P4.2 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P4.0 P4.1 P4.2 P4.3 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14 ACC15	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P4.0 P4.1 P4.2 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14 ACC15 ACC16	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14 ACC15	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC6 ACC7 ACC8 ACC9 ACC10 ACC10 ACC11 ACC12 ACC13 ACC14 ACC15 ACC16 ACC17	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM3.15 CCM4.3 CCM4.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7 P5.8	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC7 ACC8 ACC9 ACC10 ACC10 ACC11 ACC12 ACC13 ACC13 ACC14 ACC15 ACC16 ACC17 ACC18	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7 CCM4.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC6 ACC7 ACC8 ACC9 ACC10 ACC10 ACC11 ACC12 ACC13 ACC14 ACC15 ACC16 ACC17	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM3.15 CCM4.3 CCM4.7
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7 P5.8 P5.9	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC7 ACC8 ACC9 ACC10 ACC10 ACC11 ACC12 ACC13 ACC13 ACC14 ACC15 ACC16 ACC17 ACC18	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7 CCM4.11 CCM4.15
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7 P5.8 P5.9 P5.10	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC7 ACC8 ACC9 ACC10 ACC10 ACC11 ACC12 ACC13 ACC13 ACC14 ACC15 ACC16 ACC17 ACC18	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7 CCM4.11
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7 P5.8 P5.9 P5.10 P5.11	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC11 ACC12 ACC13 ACC14 ACC15 ACC16 ACC17 ACC18 ACC19 ACC20	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7 CCM4.11 CCM4.15 CCM5.3
DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.6 P3.5 P3.8 P3.9 P3.10 P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7 P5.8 P5.9 P5.10	T7R T8M T8R ACC0 ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14 ACC15 ACC16 ACC17 ACC16 ACC19	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T78CON.6 T78CON.11 T78CON.11 T78CON.14 CCM0.3 CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7 CCM4.11 CCM4.15
	DEFR DEFR DEFR DEFR DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	DEFR 0F036h DEFR 0F032h DEFR 0F030h DEFB P2.0 DEFB P2.1 DEFB P2.2 DEFB P2.3 DEFB P2.4 DEFB P2.5 DEFB P2.6 DEFB P2.7 DEFB P2.7 DEFB P2.9 DEFB P2.9 DEFB P2.10 DEFB P2.11 DEFB P2.13 DEFB P2.13 DEFB P2.13 DEFB P2.14 DEFB P2.14 DEFB P2.15 LIT 'CC010' LIT 'CC10' LIT 'CC30' DEFB P3.1 DEFB P3.4	DEFROF036hAN14DEFROF034hAN15DEFROF034hT6EUDDEFROF030hT5EUDDEFROF030hT5EUDDEFBP2.0T4EUDDEFBP2.1T2EUDDEFBP2.3P0UT0DEFBP2.4P0UT1DEFBP2.5P0UT2DEFBP2.6P0UT3DEFBP2.7CC28IODEFBP2.8CC30IODEFBP2.10CC30IODEFBP2.11CC16IODEFBP2.12CC16IODEFBP2.13CC16IODEFBP2.14CC16IODEFBP2.15CC20IOLTT'CC0IO'CC20IOLTT'CC2IO'CC22IOLTT'CC2IO'CC22IOLTT'CC2IO'CC22IODEFBP3.1T0RDEFBP3.4T1RDEFBP3.4T1R	DEFR DEFROF036hAN14DEFB DEFBDEFROF034hAN15DEFBDEFROF032hTGEUDLITDEFROF030hT5EUDLITDEFBP2.0T4EUDLITDEFBP2.1T2EUDLITDEFBP2.2P0UT0DEFBDEFBP2.3P0UT0DEFBDEFBP2.4P0UT1DEFBDEFBP2.5P0UT2DEFBDEFBP2.6P0UT3DEFBDEFBP2.7CC2810DEFBDEFBP2.10CC3110DEFBDEFBP2.11CC1610DEFBDEFBP2.12CC1610DEFBDEFBP2.13CC1710DEFBDEFBP2.14CC1610DEFBDEFBP2.15CC2110DEFBDEFBP2.14CC1810DEFBDEFBP2.15CC2310DEFBDEFBP2.14CC1910DEFBDEFBP2.15CC2110DEFBDEFBP3.1CC210DEFBDEFBP3.2T0MDEFBDEFBP3.4T1MDEFBDEFBP3.4T1RDEFBDEFBP3.4T1RDEFBDEFBP3.4T1RDEFBDEFBP3.4T1RDEFB

23:03:14		
ACC22 ACC23	DEFB DEFB	CCM5.11 CCM5.15
ACC24 ACC25	DEFB DEFB	CCM6.3 CCM6.7
ACC26	DEFB	CCM6.11
ACC27	DEFB	CCM6.15
ACC28	DEFB	CCM7.3
ACC29	DEFB DEFB	CCM7.7 CCM7.11
ACC30 ACC31	DEFB	CCM7.15
	DEFB	T2CON.6
T2R T2UD	DEFB	T2CON.7
T2UDE	DEFB	T2CON.8
T3R	DEFB	T3CON.6
T3UD	DEFB	T3CON.7
T3UDE	DEFB	T3CON.8
T3OE	DEFB	T3CON.9
T3OTL	DEFB	T3CON.10
T4R	DEFB	T4CON.6
T4UD	DEFB	T4CON.7
T4UDE	DEFB	T4CON.8
T5R	DEFB	T5CON.6
T5UD	DEFB	T5CON.7
T5UDE	DEFB	T5CON.8
T5CLR T5SC	DEFB DEFB	T5CON.14 T5CON.15
	0070070000000	
T6R		T6CON.6 T6CON.7
T6UD T6UDE	DEFB DEFB	TECON. 8
T60E	DEFB	T6CON.9
TEOTL	DEFB	T6CON.10
T6SR	DEFB	T6CON.15
T2IE	DEFB	T2IC.6
T2IR	DEFB	T2IC.7
TJIE	DEFB	T3IC.6
T3IR T4IE	DEFB DEFB	T3IC.7 T4IC.6
T4IE T4IR	DEFB	T4IC.7
T5IE	DEFB	T5IC.6
T5IR	DEFB	T5IC.7
T6IE	DEFB	T6IC.6
T6IR	DEFB	T6IC.7
CRIE	DEFB	CRIC.6
CRIR	DEFB	CRIC.7
SOTIE	DEFB	SOTIC.6
SOTIR	DEFB	SOTIC.7
SORIE	DEFB	SORIC.6
SORIR SOEIE	DEFB DEFB	SORIC.7 SOEIC.6
SOEIR	DEFB	SOEIC.0
SOTBIE	DEFB	SOTBIC.6
SOTBIR	DEFB	SOTBIC.7
SSCTIE	DEFB	SSCTIC.6
SSCTIR	DEFB	SSCTIC.7
	100002422	

99/05/08

SSCRIE

SSCRIR

SSCEIE

SSCEIR

SSCTE

SSCRE

SSCPE

SSCBE

CCOIE

CCOIR

CC1IE

CC1IR

CC2IE

CC2IR

CC3IE

CC3IR CC4IE

CC4IR

CC5IE

CC5IR

CC6IE

CC6IR

CC7IE

CC7IR

CC8IE

CC8IR

CC9IE CC9IR

CC10IE

CC10IR

CC11IE

CC11IR CC12IE

CC12IR

CC13IE

CC13IR CC14IE

CC14IR

CC15IE

CC15IR

CC16IE

CC16IR CC17IE

CC17IR

CC18IE CC18IR

CC19IE

CC19IR

CC20IE

CC20IR

CC21IE

CC21IR CC22IE

CC22IR

CC23IE CC23IR

CC24IE

CC24IR

CC25IE

CC25IR

CC26IE CC26IR

CC27IE

DEFB

DEFB

DEFB

DEFB

LIT

LIT

LIT

LIT

DEFB

SSCRIC.6

SSCRIC.7

SSCEIC.6

SSCEIC.7

'SSCTEN'

'SSCREN' 'SSCPEN'

'SSCBEN'

CCOIC.6

CCOIC.7

CC1IC.6

CC1IC.7 CC2IC.6

CC2IC.7

CC3IC.6

CC3IC.7

CC4IC.6

CC4IC.7

CC5IC.6

CC5IC.7

CC6IC.6

CC6IC.7

CC7IC.6

CC7IC.7

CC8IC.6

CC8IC.7

CC9IC.6

CC9IC.7

CC10IC.6

CC10IC.7

CC11IC.6

CC11IC.7

CC12IC.6

CC12IC.7

CC13IC.6

CC13IC.7

CC14IC.6

CC14IC.7

CC15IC.6

CC15IC.7

CC16IC.6

CC16IC.7

CC17IC.6

CC17IC.7

CC18IC.6

CC18IC.7

CC19IC.6

CC19IC.7

CC20IC.6

CC20IC.7

CC21IC.6

CC21IC.7

CC22IC.6

CC22IC.7

CC23IC.6

CC23IC.7

CC24IC.6

CC24IC.7

CC25IC.6

CC25IC.7

CC26IC.6

CC26IC.7

CC27IC.6

99/	05/	08
23:	03:	14

PWMCON0.0 PWMCON0.1 PWMCON0.2 PWMCON0.3 PWMCON0.4 PWMCON0.5 PWMCON0.6 PWMCON0.7 PWMCON0.8 PWMCON0.9 PWMCON0.10 PWMCON0.11 PWMCON0.12 PWMCON0.13 PWMCON0.14 PWMCON0.15 PWMCON1.0 PWMCON1.1 PWMCON1.2 PWMCON1.3 PWMCON1.4 PWMCON1.5 PWMCON1.6 PWMCON1.7 PWMCON1.12 PWMCON1.14 PWMCON1.15 PWMIC.6 PWMIC.7 XP3IC.6 XP3IC.7 XP2IC.6 XP2IC.7 XP1IC.6 XP1IC.7 XPOIC.6 XPOIC.7

	, jane		
	100		
	100	10	

CC27IR	DEFB	CC27IC.7		
CC28IE	DEFB	CC28IC.6	PTRO	DEFB
			PTR1	DEFB
CC28IR	DEFB	CC28IC.7		
CC29IE	DEFB	CC29IC.6	PTR2	DEFB
CC29IR	DEFB	CC29IC.7	PTR3	DEFB
CC30IE	DEFB	CC30IC.6	PTIO	DEFB
			PTI1	DEFB
CC30IR	DEFB	CC30IC.7		
CC31IE	DEFB	CC31IC.6	PTI2	DEFB
CC31IR	DEFB	CC31IC.7	PTI3	DEFB
0001111			PIEO	DEFB
			PIE1	DEFB
ADCIE	DEFB	ADCIC.6		
ADCIR	DEFB	ADCIC.7	PIE2	DEFB
ADEIE	DEFB	ADEIC.6	PIE3	DEFB
ADEIR	DEFB	ADEIC.7	PIRO	DEFB
ADEIK	DEFD	ADDIC. /	PIR1	DEFB
TOIE	DEFB	TOIC.6	PIR2	DEFB
TOIR	DEFB	TOIC.7	PIR3	DEFB
T1IE	DEFB	TIIC.6		
	DEFB	TIIC.7	PENO	DEFB
T1IR			PEN1	DEFB
T7IE	DEFB	T7IC.6		
T7IR	DEFB	T7IC.7	PEN2	DEFB
T8IE	DEFB	T8IC.6	PEN3	DEFB
T8IR	DEFB	T8IC.7	PMO	DEFB
TOIR	DEFD	1010.7	PM1	DEFB
New Standard Downline				
ADST	DEFB	ADCON.7	PM2	DEFB
ADBSY	DEFB	ADCON.8	PM3	DEFB
ADWR	DEFB	ADCON.9	PB01	DEFB
			PS2	DEFB
ADCIN	DEFB	ADCON.10		
ADCRQ	DEFB	ADCON.11	PS3	DEFB
ILLBUS	DEFB	TFR.0	PWMIE	DEFB
			PWMIR	DEFB
ILLINA	DEFB	TFR.1	FWHIK	DEFD
ILLOPA	DEFB	TFR.2		
PRTFLT	DEFB	TFR.3	XP3IE	DEFB
UNDOPC	DEFB	TFR.7	XP3IR	DEFB
			XP2IE	DEFB
STKUF	DEFB	TFR.13		
STKOF	DEFB	TFR.14	XP2IR	DEFB
NMI	DEFB	TFR.15	XP1IE	DEFB
ND425-T			XP1IR	DEFB
1 DOT 11	DDDD		XPOIE	DEFB
WDTIN	DEFB	WDTCON.0		
WDTR	DEFB	WDTCON.1	XPOIR	DEFB
SOSTP	DEFB	SOCON.3		
SOREN	DEFB	S0CON. 4		
SOPEN	DEFB	SOCON.5		
SOFEN	DEFB	SOCON.6		
SOOEN	DEFB	SOCON.7		
SOPE	DEFB	SOCON.8		
SOFE	DEFB	SOCON.9		
SOOE	DEFB	S0CON.10		
SOODD	DEFB	S0CON.12		
SOBRS	DEFB	S0CON.13		
SOLB	DEFB	SOCON.14		
SOR	DEFB	SOCON.15		
SSCHB	DEFB	SSCCON.4		
SSCPH	DEFB	SSCCON.5		
SSCPO	DEFB	SSCCON.6		
SSCTEN	DEFB	SSCCON.8		
SSCREN	DEFB	SSCCON.9		
SSCPEN	DEFB	SSCCON.10		
SSCBEN	DEFB	SSCCON.11		
SSCBSY	DEFB	SSCCON.12		
SSCMS	DEFB	SSCCON.14		
SSCEN	DEFB	SSCCON.15		

B.3 14V Bus CAN Node 2

On the next page starts the code for the 14V bus CAN node 2. The files for the node are as follows.

- 1. comp212.bat
- 2. main212.asm
- 3. cnmod212.asm
- 4. canmo212.asm
- 5. cnint212.asm
- 6. atod212.asm
- 7. tmrs212.asm
- 8. linker.lnv
- 9. Reg167b.def



al66 main212.asm al66 cnmod212.asm al66 cnmod212.asm al66 cnint212.asm al66 atod212.asm al66 tmrs212.asm al66 tmrs212.asm ll66 LINK main212.obj cnmod212.obj canmo212.obj cnint212.obj atod212.obj tmrs212.obj TO locatein.lno ll66 @linker.lnv

ihex166 -i16 locate.out -o main212.hex





main212.asm



;; Initialize CAN Bus SSEGMENTED ; Call the CAN initialization function CALL canin SEXTEND ;; End of CAN Bus Initialization SEXTSFR ; CAN USE ALL internal RAM for Stack SEXTSSK meto: \$EXTMEM NOP ; just loop here waiting \$NOMOD166 NOP \$STDNAMES(reg167b.def) JMP meto \$SYMBOLS RET ; return main ENDP NAME main ; define a common register area of 16 register mainseg ENDS RBANK1 COMREG R0-R15 startupsec SECTION CODE ; codesegment that contains reset int pointer ; default stack size of 256 Words SSKDEF 4 ; reset interrupt number is zero at Oh sysreset PROC TASK INTNO=0H ; forces next instruction to be located at Oh ORG 000H ASSUME DPP3:SYSTEM ; installs a pointer to the startup routine JMP start ; return from interrupt RETI EXTERN canin: FAR ; Can function sysreset ENDP EXTERN atod_initialize:FAR ; external atod initialization startupsec ENDS EXTERN atod_timer_initialize:FAR END mainseg SECTION CODE main PROC FAR start: DISWDT ; disable the watchdog timer BSET IEN ; Globally Enable Interrupts both global ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh EINIT ; end initialization ;; End of external memory bus initialization ;; Initialize the Data Page pointers for this section MOV DPP3, #03h ; make DPP3 point to system ;; End of Data Page Pointer Initialization ;; Make the direction of Port 2 to output MOV DP2, ONES ;; Make sure Port 2 is in push/pull mode MOV ODP2, ONES ;; Initialize The Stack ;; The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #OFBFFh the highest ;; byte that the stack pointers can point to is #0FBFEh MOV STKUN, #0FBFEh: Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP, #OFBFEh ; Set the Stack Pointer ;; End of Stack Initialization ;; Initialize the Analog to Digital Converter CALL atod_initialize; atod ;; End of A/D initialization ;; Initialize A/D timer CALL atod_timer_initialize; timers ;; End of A/D timer initialization



cnmod212.asm



	and the second	chinic d2	1 2 (dom	And the second
			RET	
\$SEGMEN \$EXTEND			canin ENDP	
\$EXTSFR			actall DDOG END	; This Procedure sets all of the Mess objs invalid
\$EXTMEM			setall PROC FAR	it counts up to 15 and initializes all of the message
\$NOMOD1	MES(reg167b.def)		;; objects along the	
\$SYMBOL			PUSH R2	
•			PUSH R4	
NAME ca	anmod		PUSH R5	
	2010000 D0 D15	define a common verification and of 16 registers	AND R5,ZEROS OR R5, #01h	; Set counter to 1 for first MO
GLOBAL	COMREG R0-R15	; define a common register area of 16 registers ; The function must be declared Global at the	AND R2, ZEROS	, set counter to i for first no
GLOBAL	canin	; beginning of the module	OR R2, #0EF10h	; Set pointer to MO1
			AND R4, ZEROS	
EXTERN	canmocfg:FAR	; configures specific Message objects	OR R4, #5555h	; Set R4 to make MObs invalid
NGGUNG	DPP3:SYSTEM		nextreg:MOV [R2],R4	; make all message objects invalid
ASSUME	DPPS:SISTEM		ADD R2, #10h	, make all medeage objects intalla
canfunc	SECTION CODE	; codesegment that contains reset int pointer	CMPI1 R5, #0Fh	
			JMPA CC_NZ, nextreg	;
canin			POP R5	
	PUSH RO		POP R4 POP R2	
	PUSH R1		RET	
	:: set all of the	e CAN control registers	setall ENDP	
		; set control register to zero		
	MOV R1, #0043h	; Set IE and INIT bits	canfunc ENDS	
	OR C1CSR,R1 ;	set control register to R1's value	END	
	AND CIETE ZEROS	; set Bit timing register to zero		
	MOV R1, #03447h	; set for 125k operation		
		set Bit timing register parameters		
		; set Global Mask short register to zero : EOFF is what DAVE initialize		
	MOV R1, #0FFFFh OR C1GMS, R1 ;			
	ok cions, ki ,			
		5 ; set Upper global mask long to zero		
	MOV R1, #0FFFFh			
	OR CIUGML, R1			
	MOV R1, #0F8FFh			
	AND CILGML, ZEROS			
	OR C1LGML, R1	; lower global mask		
	AND C1UMLM, ZEROS OR C1UMLM, R1	; upper mask of last register		
	AND C1LMLM, ZEROS			
	OR CILMLM, R1	; lower mask of last register		
	CALL setall	; sets all of the CAN registers to off		
	CALL canmocfg	; Configures specific Message Objects		
	;; Setup CAN inte	errupt and Initialize CAN module		
E	EXTR #4			
	AND XPOIC, ZEROS AND R0,ZEROS	; configure CAN interrupt control Register		
	OR R0, #0073h	; enable interrupt, level is 10 group is 2		
	OR XPOIC, RO	; Configure CAN interrupt Control Register		
	AND R1, ZEROS	success if I also the ODI accord to the DMD		
	OR R1, #00041h XOR C1CSR, R1	; crashes if I clear the CPU access to the BTR ; end initialize CAN interrupt		
	POP R1	ena inicialize ena interiapo		
	POP R0			

99/05/24 10:47:46 canmo2	12.asm
<pre>\$SEGMENTED \$EXTEND \$EXTSPR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS NAME canmo RBANK1 COMREG R0-R15 ; declare bank of 16 global registers GLOBAL canmocfg</pre>	AND R1, ZEROS OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero OR R3, #00012h ; The number is the Message ID for Message Object 5 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0038h ; put 0AAh into first data byte and set to receive MOV MCD_M5,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o
can_module SECTION CODE	MOV DATA_M5, ZEROS ; fill the data of the MO with ZEROS
ASSUME DPP3:SYSTEM canmocfg PROC FAR PUSH R1 PUSH R2 PUSH R3 ;; Now set specific CAN control Registers ;; initialize message object 1 ;; initialize message object to be invalid does or removing the code until ;; the comment "Setup CAN interrupt and Initialize" does ;; nothing to prevent the occurrance of the interrupt for the CAN system MOV R2, #MCR_M1 ; start of Message Object 1 AND R1, ZEROS OR R1, #5599h ; Generate a Receive Interrupt if this message object ac tivates MOV [R2],R1 ; set MO1's Control register ADD R2, #2h ; point to Upper Arbitration register AND R3, ZEROS ; set R3 to OR R3, #08001h ; message id for message object 1 MOV [R2],R3 ; message id = #0003h ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0300h ; put OAAh into first data byte and set to receive MOV MCD_M1,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes of data MOV DATA_M1, ZEROS ; fill the Data of the MO with Zeros	POP R3 POP R1 RET canmocfg ENDP can_module ENDS END
MOV R2, #MCR_M3 ; start of Message Object 3 AND R1, ZEROS OR R1, #5595h ; Generate a receive interrupt if this message object ac tivates MOV [R2],R1 ; set MO3'S Control register ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero OR R3, #04077h ; The number is the Message ID for Message Object 3 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0038h ; put 000h into first data byte and set to receive MOV MCD_M3,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes of da	
ta MOV DATA_M3, ZEROS ; Fill the Data of the MO with Zeros	

99/05/09 11:40:10

\$SEGMENTED
\$EXTEND
\$EXTSFR
\$EXTMEM
\$NOMOD166
\$STDNAMES(reg167b.def)
\$SYMBOLS

NAME canint RBANK1 COMREG R0-R15

; declare bank of 16 global registers

ASSUME DPP3:SYSTEM

can_interrupts SECTION CODE

can_receive_interrupt PROC TASK INTNO=040h ORG 0100h CALL can_receive_interrupt_handler RETI can_receive_interrupt ENDP

can_receive_interrupt_handler PROC FAR PUSH R0

PUSH R1 PUSH R2 MOVB RLO, INTID ; Read the CAN interrupt ID buffer CMPB RLO, #03h ; See if the interrupt came from M01 JMP cc_Z, message_one_interrupt; if interrupt from M01 handle MOV R1, #05555h MOV R2, #05599h MOV MCR_M2, R1 MOV R0, DATA_M2 MOV MCR_M2, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch MOV R2,MCR_M6

MOV MCR_M6, R1 MOV DATA_M6, R0 MOV MCR_M6, R2 CMP R0, #01h JMP cc_NZ, turn_off_heated_rear_window BSET P2.1 JMP exit_function

turn_off_heated_rear_window: CMP R0, #0800h

JMP cc_NZ, exit_function BCLR P2.1 JMP exit_function

message_one_interrupt:

MOV R1, #05555h MOV R2, #05599h MOV MCR_M1, R1 MOV R0, DATA_M1 MOV MCR_M1, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch

MOV R2, MCR_M5 MOV MCR_M5, R1

cnint212.asm



MOV DATA_M5, R0

MOV MCR_M5, R2 CMP R0, #01h JMP cc_NZ, turn_heater_off BSET P2.0 JMP exit_function

turn_heater_off: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.0 exit_function:

MOV R2, #0EFFFh

AND C1CSR, R2 POP R2 POP R1 POP R0 RET can_receive_interrupt_handler ENDP

can_interrupts ENDS END



atod212.asm





PUSH R3 PUSH R4 MOV R2, ADDAT ; This is so we can isolate the A/D channel from whi MOV RO, R2 ch the data is coming ; This is so we can isolate the A/D voltage sense va MOV R3, R2 ;; This code scales the data from the A/D by 21 to get the actual current fl owing through the BTS550P AND R3, #003FFh ; This isolates the lower ten bits of the A/D's output MOV R4, #01h ; No Scaling on the microcontroller AND R0, #0F000h ; The channel information is located in the upper nibble CMP R0, #01000h ; See if the information is coming from Channel 1 of the A/ JMP cc_Z, Rear_Seat_Heater_current MOV R0, #05555h ; This bit pattern deactives MCRs MOV R1, MCR M3 ; SAVE the Configuration of the MCR MOV MCR_M3, RO ; Kill the Message Control Register MUL R3, R4 ; This generates the acutal current value NOP MOV DATA_M3, MDL ; for real MOV MCR M3, R1 BSET T3R JMP exit routine Rear Seat Heater current: MOV R0, #05555h ; This bit pattern deactives MCRs MOV R1, MCR_M4 ; SAVE the Configuration of the MCR MOV MCR_M4, R0 ; Kill the Message Control Register MOV R0, #04h ;test code ;test code ADD P2, RO MUL R3, R4 ; This generates the actual current value NOP MOV DATA_M4, MDL ; for testing purposes MOV MCR M4, R1 exit_routine: POP R4 POP R3 POP R2 POP R1 POP RO RET atod function ENDP atod_handlers ENDS



```
$SEGMENTED
                                ; These are assembler controls
$EXTEND
$EXTSFR
$EXTMEM
$EXTINSTR
$NOMOD166
$STDNAMES(reg167b.def)
                                ; Assembler controls end here
$SYMBOLS
NAME timer_functions
ASSUME DPP3 : SYSTEM
RBANK1 COMREG R0-R15
GLOBAL atod_timer_initialize
atod_timer SECTION CODE
atod_timer_initialize PROC FAR
       MOV T3CON, #0004h
                                ; setup Core Timer T3
       MOV T3IC, #002Bh
       MOV T3, #0000h
                                ; Make the value in the counter equal to zero
       BSET T3IE
                                ; enable the timer interrupt
        BSET T3R
                                ; start the timer
        RET
atod_timer_initialize ENDP
atod_interrupt PROC TASK INTNO=023h
        ORG 08Ch
        CALL atod_timer_handler
        RETI
atod_interrupt ENDP
atod_timer_handler PROC FAR
                                ; stop the timer
        BCLR T3R
                                ; start an A/D conversion
        BSET ADST
        RET
atod_timer_handler ENDP
atod_timer ENDS
END
```



LOCATE locatein.lno (GENERAL) IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (040000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO locate.out



;********** ;** @(#)reg:		***************************************	
;** @(#)IEg. ;**	107D.del	1.10 12/18/97	
	r definiti	ons for the SAB C167	
		s all SFR names and BIT names	
		supplied to rm166 and a166 (STDNAMES control)	
No state and the		***********	
TRUE	DEFB	OFF20h.0, RW	
NODE142	DEFB	0FF20h.1, RW	
C1CSR	DEFA	0EF00h	
INTID	DEFA	0EF02h	
CIBTR	DEFA	0EF04h	
CIGMS	DEFA	0EF06h	
C1UGML	DEFA	0EF08h	
C1LGML	DEFA	0EF0Ah	
C1UMLM	DEFA	0EF0Ch	
C1LMLM	DEFA	0EF0Eh	
MCR_M1	DEFA	0EF10h	
MCR_M2	DEFA	0EF20h	
MCR_M3	DEFA	0EF30h	
MCR_M4 MCR_M5	DEFA	0EF50b	
MCR_M5 MCR_M6	DEFA DEFA	0EF50h 0EF60h	
MCR_M7	DEFA	0EF70h	
MCR_M8	DEFA	0EF80h	
MCR_M9	DEFA	0EF90h	
MCR_MA	DEFA	OEFAOh	
MCR_MB	DEFA	OEFBOh	
MCR_MC	DEFA	OEFCOh	
MCR_MD	DEFA	OEFDOh	
MCR_ME	DEFA	OEFEOh	
MCR_MF MCD_M1	DEFA	0EFF0h	
MCD_M1 MCD_M2	DEFA DEFA	0EF16h 0EF26h	
MCD_M3	DEFA	0EF36h	
MCD_M4	DEFA	0EF46h	
MCD_M5	DEFA	0EF56h	
MCD_M6	DEFA	0EF66h	
MCD_M7	DEFA	0EF76h	
MCD_M8	DEFA	0EF86h	
MCD_M9	DEFA	0EF96h	
MCD_MA MCD_MB	DEFA	0EFA6h	
MCD_MB MCD_MC	DEFA DEFA	0EFB6h 0EFC6h	
MCD_MD	DEFA	0EFD6h	
MCD_ME	DEFA	0EFE6h	
DATA_M1	DEFA	0EF18h	
DATA_M2	DEFA	0EF28h	
DATA_M3	DEFA	0EF38h	
DATA_M4	DEFA	0EF48h	
DATA_M5	DEFA	0EF58h	
DATA_M6	DEFA	0EF68h	
DATA_M7 DATA_M8	DEFA	0EF78h	
DATA_M8 DATA_M9	DEFA DEFA	0EF88h 0FF98b	
DATA_MA	DEFA	0EF98h 0EFA8h	
DATA_MB	DEFA	0EFB8h	
DATA_MC	DEFA	0EFC8h	
DATA_MD	DEFA	0EFD8h	
DATA_ME	DEFA	0EFE8h	
000	0005		
OP8	DEFR	0FFD6h	

P8DEFROFFD4hDP7DEFROFFD4hP7DEFROFFCEhP6DEFROFFCEhP6DEFROFFCChDP4DEFROFFCAhP4DEFROFFC6hP3DEFROFFC6hP3DEFROFFC6hP4DEFROFFC6hP3DEFROFFC6hP3DEFROFFC6hP4DFROFFC6hP5DEFROFFC6hP6OFFC6hDFRP7DEFROFFC6hSOCONDEFROFF2hSOCONDEFROFF2hSOCONDEFROFF2hADCONDEFROFFAChP5DEFROFFAChT1ICDEFROFF9hADCICDEFROFF9hADCICDEFROFF9hCC15ICDEFROFF9hCC15ICDEFROFF9hCC11ICDEFROFF8hCC10ICDEFROFF8hCC7ICDEFROFF8hCC7ICDEFROFF8hCC7ICDEFROFF8hCC1CDEFROFF8hCC1CDEFROFF7hSCEICDEFROFF7hSCEICDEFROFF7hSOEICDEFROFF6hSCTICDEFROFF6hSCTICDEFROFF6hSOTICDEFROFF6hSCTICDEFROFF6hSCTICDEFROFF6hSOTIC <t< th=""><th></th><th></th><th></th></t<>			
P7 DEFR OFFD0h DP6 DEFR OFFCEh P6 DEFR OFFCEh DP4 DEFR OFFCAh P4 DEFR OFFCAh DP3 DEFR OFFCAh DP2 DEFR OFFCAh DP2 DEFR OFFCAh SCCON DEFR OFFCAh SSCCON DEFR OFFCAh SOCON DEFR OFFACh SSCCON DEFR OFFACh P5 DEFR OFFACh P5 DEFR OFFACh ADCON DEFR OFFACh ADCIC DEFR OFFACh ADCIC DEFR OFF9Ch ADCIC DEFR OFF9Ah CC11C DEFR OFF9Ah CC11C DEFR OFF9Ah CC11C DEFR OFF9Ah CC11C DEFR OFF8Ah CC3IC DEFR OFF8Ah CC3IC <td< td=""><td></td><td>DEFR</td><td>0FFD4h</td></td<>		DEFR	0FFD4h
DP6DEFROFFCEhP6DEFROFFCChDP4DEFROFFCChP4DEFROFFCChP3DEFROFFCChP3DEFROFFCChP2DEFROFFCChSSCCONDEFROFFB2hSOCONDEFROFFB2hSOCONDEFROFFB2hSOCONDEFROFFAChP5DEFROFFAChP5DEFROFFAChP5DEFROFFAChADCONDEFROFFAChADCICDEFROFF92hADCICDEFROFF98hCC15ICDEFROFF96hCC14ICDEFROFF96hCC11ICDEFROFF96hCC11ICDEFROFF8hCC3ICDEFROFF8hCC4ICDEFROFF8hCC3ICDEFROFF8hCC3ICDEFROFF8hCC1CDEFROFF7hCC1CDEFROFF7hSCTICDEFROFF7hSOCICDEFROFF7hSOCICDEFROFF6hSOTICDEFROFF6hSOTICDEFROFF6hSOTICDEFROFF6hT4ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT4ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6h<		DEFR	0FFD2h
P6DEFROFFCChDP4DEFROFFCChP4DEFROFFCChP3DEFROFFCChP3DEFROFFCChP2DEFROFFCChSSCCONDEFROFFCChSSCCONDEFROFFBChSOCONDEFROFFBChWDTCONDEFROFFAChP5DEFROFFAChP5DEFROFFAChADCICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF96hCC15ICDEFROFF96hCC14ICDEFROFF96hCC11ICDEFROFF8hCC10ICDEFROFF8hCC10ICDEFROFF8hCC10ICDEFROFF8hCC10ICDEFROFF8hCC10ICDEFROFF8hCC10ICDEFROFF8hCC1CDEFROFF8hCC1CDEFROFF8hCC1CDEFROFF8hCC1CDEFROFF7hCC1CDEFROFF7hSSCRICDEFROFF7hSORICDEFROFF6hSOTICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6h	P7	DEFR	OFFDOh
DP4 DEFR OFFCAh P4 DEFR OFFCAh P3 DEFR OFFCAh DP3 DEFR OFFCAh DP2 DEFR OFFCAh DP2 DEFR OFFCAh P2 DEFR OFFCAh SOCON DEFR OFFACh SSCCON DEFR OFFACh SOCON DEFR OFFACh P5 DEFR OFFACh P5 DEFR OFFACh ADCON DEFR OFF9Ch ADCIC DEFR OFF9Ch C14IC DEFR OFF9Ch C21C DEFR OFF8Ch C21C DEFR OFF8Ch C21C D		DEFR	
P4DEFROFFC8hDP3DEFROFFC6hP3DEFROFFC6hP2DEFROFFC2hP2DEFROFFC0hSSCCONDEFROFF2hSOCONDEFROFF2hSOCONDEFROFF2hSOCONDEFROFF2hSOCONDEFROFFAchTFRDEFROFFAchP5DEFROFFAchADCONDEFROFFAchTICDEFROFF9chADCICDEFROFF9chADCICDEFROFF9chADCICDEFROFF9chADCICDEFROFF9chC15ICDEFROFF9chC11CDEFROFF9chC11CDEFROFF8chC211CDEFROFF8chC211CDEFROFF8chC211CDEFROFF8chC211CDEFROFF8chC211CDEFROFF8chC211CDEFROFF8chC21CDEFROFF8chC21CDEFROFF8chC21CDEFROFF7chSSCEICDEFROFF7chSSCEICDEFROFF7chSOEICDEFROFF6chSCTICDEFROFF6chSCTICDEFROFF6chSOTICDEFROFF6chT5ICDEFROFF6chT5ICDEFROFF6chT5ICDEFROFF6chT5ICDEFROFF6chT5ICDEFR <td></td> <td></td> <td></td>			
DP3DEFROFFC6hP3DEFROFFC6hP2DEFROFFC6hP2DEFROFFC6hSSCCONDEFROFFB2hSOCONDEFROFFB2hSOCONDEFROFFB2hSOCONDEFROFFB2hSOCONDEFROFFAChTFRDEFROFFAChP5DEFROFFA2hADCONDEFROFFA2hADCICDEFROFF92hTOICDEFROFF92hADCICDEFROFF94hADCICDEFROFF94hCC15ICDEFROFF94hCC15ICDEFROFF94hCC11ICDEFROFF94hCC11ICDEFROFF84hCC10ICDEFROFF84hCC7ICDEFROFF84hCC6ICDEFROFF84hCC3ICDEFROFF84hCC3ICDEFROFF76hSCCICDEFROFF76hSCCICDEFROFF74hSSCTICDEFROFF74hSOTICDEFROFF66hT5ICDEFROFF66hT4ICDEFROFF66hT4ICDEFROFF66hT4ICDEFROFF66hT4ICDEFROFF66hT4ICDEFROFF66hT5ICDEFROFF66hT4ICDEFROFF66hT5ICDEFROFF66hT4ICDEFROFF66hT5ICDEFROFF66hT4IC <t< td=""><td></td><td></td><td></td></t<>			
P3DEFROFFC4hDP2DEFROFFC2hP2DEFROFFC2hP2DEFROFFC2hSSCCONDEFROFFB2hSOCONDEFROFFB2hSOCONDEFROFFAChTFRDEFROFFAChP5DEFROFFAChT1ICDEFROFFAChT1ICDEFROFF9AhADCONDEFROFF9AhADCICDEFROFF9AhADCICDEFROFF9AhADCICDEFROFF9AhCC15ICDEFROFF9AhCC12ICDEFROFF9AhCC12ICDEFROFF9AhCC11ICDEFROFF9AhCC11ICDEFROFF8AhCC3ICDEFROFF8AhCC4ICDEFROFF8AhCC4ICDEFROFF8AhCC4ICDEFROFF8AhCC3ICDEFROFF7AhCC1ICDEFROFF7AhCC1ICDEFROFF7AhCC1ICDEFROFF7AhCC1ICDEFROFF7AhSCCTICDEFROFF7AhSOCICDEFROFF7AhSOCICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5IC			
DP2DEFROFFC2hP2DEFROFFC2hSOCONDEFROFF2ChSOCONDEFROFF2ChSOCONDEFROFFAChTFRDEFROFFAChP5DEFROFFAChP5DEFROFF3ChADCONDEFROFF9ChADCONDEFROFF9ChADCICDEFROFF9ChADEICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF9ChCC15ICDEFROFF9ChCC13ICDEFROFF9ChCC11ICDEFROFF8ChCC10ICDEFROFF8ChCC6ICDEFROFF8ChCC6ICDEFROFF8ChCC6ICDEFROFF8ChCC6ICDEFROFF8ChCC3ICDEFROFF8ChCC1ICDEFROFF7ChCC1ICDEFROFF7ChCC1ICDEFROFF7ChCC1ICDEFROFF7AhSCEICDEFROFF7AhSCEICDEFROFF7AhSORICDEFROFF6AhT6ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT6ICDEFROFF6AhT5ICDEFROFF6AhT6ICDEFROFF6AhT6ICDEFROFF6AhT5ICDEFROFF6AhT6ICDEFROFF6AhT6IC			
P2DEFROFFCOhSSCCONDEFROFF2hSOCONDEFROFF2hSOCONDEFROFF2hWDTCONDEFROFFAchP5DEFROFFAchP5DEFROFF2hADCONDEFROFF2hADCONDEFROFF2hADCONDEFROFF9chADCICDEFROFF9chADEICDEFROFF9chADCICDEFROFF9chADCICDEFROFF9chCC15ICDEFROFF9chCC12ICDEFROFF9chCC12ICDEFROFF8chCC10ICDEFROFF8chCC9ICDEFROFF8chCC6ICDEFROFF8chCC5ICDEFROFF8chCC4ICDEFROFF8chCC3ICDEFROFF7chCC1ICDEFROFF7chCC1ICDEFROFF7chCC1ICDEFROFF7chSCEICDEFROFF7chSCEICDEFROFF7chSORICDEFROFF6chSORICDEFROFF6chT4ICDEFROFF6chT5ICDEFROFF6chT4ICDEFROFF6chT5ICDEFROFF6chT4ICDEFROFF6chT5ICDEFROFF6chT5ICDEFROFF6chT5ICDEFROFF6chT4ICDEFROFF6chT5ICDEFROFF6chT4IC			
SSCCONDEFROFFB2hSOCONDEFROFFB2hSOCONDEFROFFB2hTFRDEFROFFA2hADCONDEFROFFA2hADCONDEFROFFA2hADCONDEFROFFA2hADCONDEFROFFA2hADCONDEFROFFA2hADCICDEFROFF92hTIICDEFROFF92hADCICDEFROFF94hCC15ICDEFROFF94hCC13ICDEFROFF92hCC12ICDEFROFF92hCC11ICDEFROFF86hCC10ICDEFROFF88hCC7ICDEFROFF88hCC7ICDEFROFF86hCC6ICDEFROFF86hCC4ICDEFROFF86hCC3ICDEFROFF76hSSCRICDEFROFF76hSSCRICDEFROFF76hSOCICDEFROFF76hSOCICDEFROFF6hSOTICDEFROFF6hSOTICDEFROFF6hT3ICDEFROFF6hT4ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT4ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5IC <td< td=""><td></td><td></td><td></td></td<>			
SOCONDEFROFFBOHWDTCONDEFROFFAChTFRDEFROFFAChP5DEFROFFAChADCONDEFROFFAChTIICDEFROFFAChTIICDEFROFFAChADEICDEFROFF9ChADEICDEFROFF9AhADCICDEFROFF9AhADCICDEFROFF9AhADCICDEFROFF9AhCC15ICDEFROFF9AhCC13ICDEFROFF9AhCC11ICDEFROFF9AhCC10ICDEFROFF8AhCC7ICDEFROFF8AhCC7ICDEFROFF8AhCC6ICDEFROFF8AhCC7ICDEFROFF8AhCC4ICDEFROFF8AhCC3ICDEFROFF7AhCC1ICDEFROFF7AhCC1ICDEFROFF7AhCC1ICDEFROFF7AhSCRICDEFROFF7AhSOCICDEFROFF7AhSOCICDEFROFF6AhSOTICDEFROFF6AhSOTICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5			
WDTCONDEFROFFAEhTFRDEFROFFAChP5DEFROFFAChADCONDEFROFFAChADCONDEFROFF9ChTOICDEFROFF9EhTOICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF9ChCC13ICDEFROFF9ChCC12ICDEFROFF9ChCC10ICDEFROFF8ChCC9ICDEFROFF8ChCC9ICDEFROFF8ChCC7ICDEFROFF8ChCC6ICDEFROFF8ChCC4ICDEFROFF8ChCC4ICDEFROFF8ChCC4ICDEFROFF8ChCC2ICDEFROFF7ChSSCICDEFROFF7ChSSCRICDEFROFF7ChSORICDEFROFF7ChSORICDEFROFF6ChCRICDEFROFF6ChSORICDEFROFF6ChSORICDEFROFF6ChTAICDEFROFF6ChTAICDEFROFF6ChTAICDEFROFF6ChCCM1DEFROFF3hCCM2DEFROFF6ChTACONDEFROFF4hTSCONDEFROFF4hTSCONDEFROFF4hTSCONDEFROFF4hTSCONDEFROFF4hTSCONDEFROFF4hTSCONDEFROFF4hTSCON<			
TFRDEFROFFAChP5DEFROFFAChADCONDEFROFFAChADCONDEFROFF9ChADEICDEFROFF9ChADEICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF9ChADCICDEFROFF9ChCC15ICDEFROFF9ChCC13ICDEFROFF9ChCC11ICDEFROFF8ChCC0ICDEFROFF8ChCC6ICDEFROFF8ChCC6ICDEFROFF8ChCC4ICDEFROFF8ChCC4ICDEFROFF8ChCC3ICDEFROFF7ChCC1ICDEFROFF7ChCC1ICDEFROFF7ChSCEICDEFROFF7ChSCEICDEFROFF7ChSORICDEFROFF7ChSORICDEFROFF6ChT5ICDEFROFF6ChT5ICDEFROFF6ChT5ICDEFROFF6ChT4ICDEFROFF6ChT5ICDEFROFF6ChT4ICDEFROFF52hT5ICDEFROFF6ChT6CNDEFROFF52hT6CONDEFROFF52hT6CONDEFROFF6ChT6CNDEFROFF6ChT6CNDEFROFF2hT6CONDEFROFF2hT6CO			
P5DEFROFFA2hADCONDEFROFFA0hT1ICDEFROFF90hT0ICDEFROFF92hADEICDEFROFF97hADCICDEFROFF98hCC15ICDEFROFF97hADCICDEFROFF97hADCICDEFROFF97hADCICDEFROFF97hCC15ICDEFROFF97hCC12ICDEFROFF92hCC12ICDEFROFF82hCC10ICDEFROFF8hCC9ICDEFROFF8hCC5ICDEFROFF8hCC4ICDEFROFF8hCC3ICDEFROFF80hCC3ICDEFROFF7hCC1ICDEFROFF7hCC1ICDEFROFF7hSSCEICDEFROFF7hSOEICDEFROFF7hSOEICDEFROFF6hSOTICDEFROFF6hSOTICDEFROFF6hSOTICDEFROFF6hT5ICDEFROFF6hT4ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT4ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hT5ICDEFROFF6hCCM3DEFROF			
ADCONDEFROFFAOhT1ICDEFROFF9EhT0ICDEFROFF9EhADEICDEFROFF9AhADCICDEFROFF9AhADCICDEFROFF9AhADCICDEFROFF9AhADCICDEFROFF9AhCC15ICDEFROFF9AhCC12ICDEFROFF9AhCC11ICDEFROFF9ChCC11ICDEFROFF8AhCC0ICDEFROFF8AhCC7ICDEFROFF8AhCC6ICDEFROFF8AhCC4ICDEFROFF8AhCC3ICDEFROFF8AhCC2ICDEFROFF7AhCC1ICDEFROFF7AhCC1ICDEFROFF7AhCC1ICDEFROFF7AhSCCICDEFROFF7AhSCCICDEFROFF7AhSOEICDEFROFF7AhSOEICDEFROFF6AhSORICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF5AhCCM2DEFROFF5AhCCM3DEFROFF5AhCCM1DEFROFF5AhCC			
T1IC DEFR OFF9Eh T0IC DEFR OFF9Eh T0IC DEFR OFF9Ch ADEIC DEFR OFF9Ch ADCIC DEFR OFF96h CC15IC DEFR OFF96h CC14IC DEFR OFF96h CC12IC DEFR OFF92h CC12IC DEFR OFF8Eh CC10IC DEFR OFF8Eh CC10IC DEFR OFF8Eh CC20IC DEFR OFF8Eh CC6IC DEFR OFF8Eh CC4IC DEFR OFF8Eh CC4IC DEFR OFF8Eh CC2IC DEFR OFF8Eh CC2IC DEFR OFF7Eh CC2IC DEFR OFF7Eh CC2IC DEFR OFF7Eh CC2IC DEFR OFF7Eh SCRIC DEFR OFF7Eh SCRIC DEFR OFF7Eh SORIC DEFR OFF6Eh			
TOIC DEFR OFF9Ch ADEIC DEFR OFF9Ah ADCIC DEFR OFF9Ah ADCIC DEFR OFF9Ah CC15IC DEFR OFF9Ah CC13IC DEFR OFF9Ah CC13IC DEFR OFF9Ah CC12IC DEFR OFF9Ah CC11IC DEFR OFF8Ch CC9IC DEFR OFF8Ah CC6IC DEFR OFF8Ah CC6IC DEFR OFF8Ah CC6IC DEFR OFF8Ah CC4IC DEFR OFF8Ah CC2IC DEFR OFF7Ah CC2IC DEFR OFF7Ah CC0IC DEFR OFF7Ah CC0IC DEFR OFF7Ah SCEIC DEFR OFF7Ah SCRIC DEFR OFF7Ah SORIC DEFR OFF6Ah SORIC DEFR OFF6Ah T5IC DEFR OFF6Ah			
ADEIC DEFR OFF9Ah ADCIC DEFR OFF98h CC15IC DEFR OFF96h CC14IC DEFR OFF92h CC12IC DEFR OFF92h CC12IC DEFR OFF92h CC12IC DEFR OFF92h CC12IC DEFR OFF82h CC10IC DEFR OFF8ch CC9IC DEFR OFF8ch CC6IC DEFR OFF8ch CC5IC DEFR OFF8ch CC4IC DEFR OFF8ch CC2IC DEFR OFF8ch CC2IC DEFR OFF8ch CC1C DEFR OFF7ch CC1C DEFR OFF7ch SCEIC DEFR OFF7ch SOEIC DEFR OFF7ch SORIC DEFR OFF6ch SOTIC DEFR OFF6ch T5IC DEFR OFF6ch T5IC DEFR OFF6ch			
ADCIC DEFR OFF98h CC15IC DEFR OFF96h CC15IC DEFR OFF96h CC14IC DEFR OFF94h CC12IC DEFR OFF90h CC11IC DEFR OFF86h CC01C DEFR OFF88h CC9IC DEFR OFF88h CC7IC DEFR OFF88h CC7IC DEFR OFF86h CC6IC DEFR OFF86h CC4IC DEFR OFF86h CC3IC DEFR OFF76h CC1C DEFR OFF76h SSCEIC DEFR OFF76h SSCRIC DEFR OFF6h SOTIC DEFR OFF6h T3IC DEFR OFF6h T4IC DEFR OFF6h T2I			
CC15IC DEFR OFF96h CC14IC DEFR OFF94h CC13IC DEFR OFF94h CC12IC DEFR OFF96h CC11IC DEFR OFF86h CC10IC DEFR OFF86h CC9IC DEFR OFF86h CC6IC DEFR OFF86h CC6IC DEFR OFF86h CC4IC DEFR OFF86h CC4IC DEFR OFF86h CC2IC DEFR OFF82h CC2IC DEFR OFF72h CC2IC DEFR OFF77h CC1IC DEFR OFF78h SSCEIC DEFR OFF74h SSCTIC DEFR OFF6h SOTIC DEFR OFF6h SOTIC DEFR OFF6h SOTIC DEFR OFF6h T3IC DEFR OFF6h T3IC DEFR OFF6h T3IC DEFR OFF6h T			
CC141C DEFR OFF94h CC13IC DEFR OFF92h CC12IC DEFR OFF92h CC11IC DEFR OFF8ch CC01IC DEFR OFF8ch CC9IC DEFR OFF8ch CC9IC DEFR OFF8ch CC7IC DEFR OFF8ch CC6IC DEFR OFF8ch CC5IC DEFR OFF8ch CC2IC DEFR OFF7ch CC1IC DEFR OFF7ch CC1IC DEFR OFF7ch CC1C DEFR OFF7ch SCEIC DEFR OFF7ch SCEIC DEFR OFF7ch SOEIC DEFR OFF7ch SORIC DEFR OFF6ch SOTIC DEFR OFF6ch SOTIC DEFR OFF6ch SOTIC DEFR OFF6ch T5IC DEFR OFF6ch T5IC DEFR OFF6ch <			
CC13IC DEFR OFF92h CC12IC DEFR OFF90h CC11IC DEFR OFF80h CC10IC DEFR OFF82h CC9IC DEFR OFF88h CC8IC DEFR OFF88h CC7IC DEFR OFF82h CC6IC DEFR OFF82h CC4IC DEFR OFF82h CC4IC DEFR OFF82h CC4IC DEFR OFF72h CC1IC DEFR OFF77h CC1IC DEFR OFF77h SCEIC DEFR OFF77h SOEIC DEFR OFF76h SORIC DEFR OFF62h SOTIC DEFR OFF64h T5IC DEFR OFF64h T3IC DEFR OFF64h T3IC DEFR OFF64h T3IC DEFR OFF64h T3IC DEFR OFF58h CCM2 DEFR OFF54h C			
CC121C DEFR OFF90h CC111C DEFR OFF82h CC101C DEFR OFF82h CC91C DEFR OFF88h CC91C DEFR OFF88h CC71C DEFR OFF88h CC71C DEFR OFF86h CC61C DEFR OFF86h CC51C DEFR OFF80h CC31C DEFR OFF76h CC21C DEFR OFF76h SCC1C DEFR OFF76h SSCEIC DEFR OFF76h SSCTIC DEFR OFF76h SOEIC DEFR OFF6h SOTIC DEFR OFF6ch CRC DEFR OFF6h SOTIC DEFR OFF6h SOTIC DEFR OFF6h SOTIC DEFR OFF6h T1C DEFR OFF6h T2IC DEFR OFF6h T2IC DEFR OFF6h CCM2 <td></td> <td></td> <td></td>			
CC111CDEFROFF8EhCC101CDEFROFF8ChCC91CDEFROFF8ChCC81CDEFROFF86hCC61CDEFROFF86hCC61CDEFROFF8ChCC41CDEFROFF8ChCC41CDEFROFF72hCC21CDEFROFF77hCC11CDEFROFF77hCC11CDEFROFF77hSSCRICDEFROFF76hSSCTICDEFROFF76hSOTICDEFROFF76hSOTICDEFROFF76hSOTICDEFROFF6hT51CDEFROFF6hT41CDEFROFF6hT31CDEFROFF6hT21CDEFROFF6hCCM3DEFROFF56hCCM1DEFROFF56hCCM1DEFROFF56hCCM1DEFROFF4hT5CONDEFROFF4hT5CONDEFROFF4hT5CONDEFROFF4hT5CONDEFROFF4hT5CONDEFROFF4hT2CONDEFROFF4hT2CONDEFROFF4hT2CONDEFROFF4hT2CONDEFROFF4hT2CONDEFROFF3hCCM6DEFROFF2hCCM5DEFROFF2h			
CC10ICDEFROFF8ChCC9ICDEFROFF8AhCC9ICDEFROFF8AhCC7ICDEFROFF8AhCC7ICDEFROFF8AhCC5ICDEFROFF8AhCC5ICDEFROFF8AhCC4ICDEFROFF7AhCC2ICDEFROFF7AhCC1ICDEFROFF7AhCC0ICDEFROFF7AhSCEICDEFROFF7AhSCEICDEFROFF7AhSOEICDEFROFF7AhSORICDEFROFF7AhSORICDEFROFF6AhT6ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT5ICDEFROFF6AhT2ICDEFROFF6AhT2ICDEFROFF6AhT01CONDEFROFF5AhCCM2DEFROFF5AhCCM0DEFROFF5AhT5CONDEFROFF5AhT6CONDEFROFF4AhT5CONDEFROFF4AhT5CONDEFROFF4AhT3CONDEFROFF4AhT3CONDEFROFF4AhT4CONDEFROFF4AhT5CONDEFROFF4AhT2CONDEFROFF4AhT2CONDEFROFF4AhT2CONDEFROFF4AhT4CONDEFROFF4AhT4CONDEFROFF4AhT5CONDEFROFF4AhT4CONDEFROFF4Ah			
CC9ICDEFROFF8AhCC8ICDEFROFF8AhCC8ICDEFROFF8AhCC6ICDEFROFF8AhCC5ICDEFROFF8AhCC4ICDEFROFF8AhCC4ICDEFROFF7AhCC2ICDEFROFF7AhCC0ICDEFROFF7AhCC0ICDEFROFF7AhSSCEICDEFROFF7AhSOEICDEFROFF7AhSOEICDEFROFF7AhSORICDEFROFF7AhSORICDEFROFF7AhSOTICDEFROFF6AhT6ICDEFROFF6AhT5ICDEFROFF6AhT2ICDEFROFF6AhT2ICDEFROFF6AhT2ICDEFROFF6AhT2ICDEFROFF6AhT01CONDEFROFF5AhCCM2DEFROFF5AhCCM0DEFROFF4AhT3CONDEFROFF4AhT3CONDEFROFF4AhT3CONDEFROFF4AhT3CONDEFROFF4AhT3CONDEFROFF4AhT3CONDEFROFF4AhT4CONDEFROFF4AhT5CONDEFROFF4AhT2CONDEFROFF4AhT2CONDEFROFF3AhCCM5DEFROFF2Ah			
CC8IC DEFR OFF88h CC7IC DEFR OFF86h CC6IC DEFR OFF84h CC5IC DEFR OFF82h CC4IC DEFR OFF82h CC3IC DEFR OFF72h CC2IC DEFR OFF77h CC1IC DEFR OFF78h SSCEIC DEFR OFF76h SSCRIC DEFR OFF77h SOEIC DEFR OFF76h SORIC DEFR OFF6h SOTIC DEFR OFF6h SOTIC DEFR OFF6h SOTIC DEFR OFF6h STIC DEFR OFF6h T1C DEFR OFF6h T2IC DEFR OFF6h T2IC DEFR OFF5h CCM2 DEFR OFF5h CCM3 DEFR OFF5h CCM0 DEFR OFF5h CCM0 DEFR OFF4h T3CON			
CC7IC DEFR OFF86h CC6IC DEFR OFF86h CC6IC DEFR OFF82h CC4IC DEFR OFF80h CC3IC DEFR OFF7Eh CC2IC DEFR OFF7Eh CC2IC DEFR OFF7Ah CC0IC DEFR OFF7Ah SSCEIC DEFR OFF74h SSCTIC DEFR OFF74h SOCIC DEFR OFF7Ah SOTIC DEFR OFF6h SOTIC DEFR OFF6h SOTIC DEFR OFF6h SOTIC DEFR OFF6h T1C DEFR OFF6h T4IC DEFR OFF6h T2IC DEFR OFF6h T2IC DEFR OFF5h CCM2 DEFR OFF5h CCM3 DEFR OFF5h CCM1 DEFR OFF4h T5CON DEFR OFF4h T5CON			
CC6IC DEFR OFF84h CC5IC DEFR OFF82h CC4IC DEFR OFF82h CC3IC DEFR OFF72h CC2IC DEFR OFF72h CC2IC DEFR OFF72h CC1IC DEFR OFF74h SSCEIC DEFR OFF74h SSCRIC DEFR OFF74h SSCRIC DEFR OFF74h SORIC DEFR OFF72h SORIC DEFR OFF64h SORIC DEFR OFF62h SORIC DEFR OFF64h T5IC DEFR OFF64h T5IC DEFR OFF64h T3IC DEFR OFF64h T3IC DEFR OFF64h T3IC DEFR OFF54h CCM3 DEFR OFF54h CCM1 DEFR OFF54h CCM0 DEFR OFF44h T5CON DEFR OFF46h T4CO			
CC5ICDEFROFF82hCC4ICDEFROFF80hCC3ICDEFROFF76hCC2ICDEFROFF77hCC1ICDEFROFF78hSSCEICDEFROFF76hSSCRICDEFROFF74hSOEICDEFROFF77hSOEICDEFROFF77hSORICDEFROFF76hSOTICDEFROFF66hSOTICDEFROFF66hT6ICDEFROFF66hT5ICDEFROFF66hT2ICDEFROFF66hT2ICDEFROFF66hCCM3DEFROFF58hCCM2DEFROFF56hCCM1DEFROFF56hCCM1DEFROFF56hT6CONDEFROFF56hT6CONDEFROFF46hT5CONDEFROFF46hT4CONDEFROFF46hT4CONDEFROFF46hT4CONDEFROFF40hPWMCON1DEFROFF40hPWMCON1DEFROFF32hPWMCON0DEFROFF32hCCM5DEFROFF24h			
CC4IC DEFR OFF80h CC3IC DEFR OFF7Eh CC3IC DEFR OFF7Eh CC1IC DEFR OFF7Ah CC0IC DEFR OFF7Ah CC0IC DEFR OFF7Ah SSCEIC DEFR OFF7Ah SSCRIC DEFR OFF7Ah SOEIC DEFR OFF7Ah SOEIC DEFR OFF7Ah SORIC DEFR OFF7Ah SORIC DEFR OFF6Ah STIC DEFR OFF6Ah T6IC DEFR OFF6Ah T5IC DEFR OFF6Ah T2IC DEFR OFF6Ah T2IC DEFR OFF6Ah CCM3 DEFR OFF5Ah CCM1 DEFR OFF5Ah CCM2 DEFR OFF5Ah CCM3 DEFR OFF5Ah CCM0 DEFR OFF5Ah CCM0 DEFR OFF4Ah T3CON <td></td> <td></td> <td></td>			
CC3 IC DEFR OFF7Eh CC2 IC DEFR OFF7Ch CC1 IC DEFR OFF7Ch CC0 IC DEFR OFF76h SSCEIC DEFR OFF76h SSCRIC DEFR OFF76h SSCRIC DEFR OFF76h SSCRIC DEFR OFF76h SOCIC DEFR OFF76h SORIC DEFR OFF6Ch SORIC DEFR OFF6Ch SORIC DEFR OFF6Ch CRIC DEFR OFF6Ch CRIC DEFR OFF6Ch TSIC DEFR OFF6Ch T4IC DEFR OFF6Ch T2IC DEFR OFF62h T2IC DEFR OFF62h CCM2 DEFR OFF62h CCM3 DEFR OFF56h CCM1 DEFR OFF52h CCM0 DEFR OFF4bh T5CON DEFR OFF4bh T			
CC21C DEFR OFF7Ch CC1IC DEFR OFF7Ah CC0IC DEFR OFF7Ah SSCEIC DEFR OFF7Ah SSCRIC DEFR OFF7Ah SSCRIC DEFR OFF7Ah SSCRIC DEFR OFF7Ah SSCRIC DEFR OFF7Ah SORIC DEFR OFF7Ah SORIC DEFR OFF7Ah SORIC DEFR OFF6Ah SORIC DEFR OFF6Ah TGIC DEFR OFF6Ah T5IC DEFR OFF6Ah T5IC DEFR OFF6Ah T3IC DEFR OFF6Ah T2IC DEFR OFF6Ah CCM3 DEFR OFF5Ah CCM1 DEFR OFF5Ah CCM2 DEFR OFF5Ah CCM0 DEFR OFF4Ah T5CON DEFR OFF4Ah T5CON DEFR OFF4Ah T3C		100 C	
CC11CDEFROFF7AhCC0ICDEFROFF78hSSCEICDEFROFF78hSSCRICDEFROFF74hSSCTICDEFROFF72hS0EICDEFROFF72hS0RICDEFROFF62hSOTICDEFROFF68hT6ICDEFROFF68hT5ICDEFROFF66hT4ICDEFROFF66hT4ICDEFROFF66hT2ICDEFROFF67hCCM3DEFROFF56hCCM1DEFROFF56hCCM1DEFROFF56hT6C0NDEFROFF50hT6C0NDEFROFF44hT3CONDEFROFF46hT4CONDEFROFF46hT4CONDEFROFF46hT4CONDEFROFF40hPWMCON1DEFROFF40hPWMCON1DEFROFF32hCM6DEFROFF28hCCM5DEFROFF24h			
CCOIC DEFR OFF78h SSCEIC DEFR OFF76h SSCEIC DEFR OFF76h SSCRIC DEFR OFF72h SOEIC DEFR OFF70h SORIC DEFR OFF70h SORIC DEFR OFF6Eh SOTIC DEFR OFF6Ah T6IC DEFR OFF6Ah T5IC DEFR OFF6Ah T3IC DEFR OFF6Ah T2IC DEFR OFF6Ah T2IC DEFR OFF6Ah CCM3 DEFR OFF5Ah CCM2 DEFR OFF5Ah CCM3 DEFR OFF5Ah CCM0 DEFR OFF5Ah CCM0 DEFR OFF5Ah CCM0 DEFR OFF5Ah T5CON DEFR OFF4Ah T5CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON </td <td></td> <td></td> <td></td>			
SSCEIC DEFR OFF76h SSCRIC DEFR OFF74h SSCRIC DEFR OFF72h SOEIC DEFR OFF70h SORIC DEFR OFF6Eh SOTIC DEFR OFF6Ch CRIC DEFR OFF6Ch CRIC DEFR OFF6Ah T6IC DEFR OFF6Ah T5IC DEFR OFF6Ah T3IC DEFR OFF6Ah T2IC DEFR OFF6Ah CCM2 DEFR OFF6Ah CCM3 DEFR OFF5Ah CCM1 DEFR OFF5Ah CCM0 DEFR OFF5Ah CCM0 DEFR OFF4Ah T5CON DEFR OFF4Ah T5CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON </td <td></td> <td></td> <td></td>			
SSCRIC DEFR OFF74h SSCTIC DEFR OFF72h SOEIC DEFR OFF70h SORIC DEFR OFF6Eh SOTIC DEFR OFF6Ah TGIC DEFR OFF6Ah TJIC DEFR OFF6Ah TJIC DEFR OFF6Ah CCM3 DEFR OFF5Ah CCM1 DEFR OFF5Ah CCM0 DEFR OFF4Ah T5CON DEFR OFF4Ah T5CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3C			
SOEIC DEFR OFF70h SORIC DEFR OFF6Eh SOTIC DEFR OFF6Eh CRIC DEFR OFF6Ah T6IC DEFR OFF6Ah T5IC DEFR OFF6Ah T3IC DEFR OFF6Ah T2IC DEFR OFF6Dh CCM3 DEFR OFF5Ah CCM2 DEFR OFF5Ah CCM0 DEFR OFF5Ah T01CON DEFR OFF5Ah T5CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON DEFR OFF4Ah T3CON DEFR OFF4Ah PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF32h PWMCON0 DEFR OFF32h CCM6 DEFR OFF2Ah	SSCRIC	DEFR	
SORIC DEFR OFF6Eh SOTIC DEFR OFF6Ch CRIC DEFR OFF6Ah T6IC DEFR OFF6Ah T5IC DEFR OFF6Ah T4IC DEFR OFF6Ah T2IC DEFR OFF6Ah T2IC DEFR OFF6Ah CCM3 DEFR OFF5Ah CCM2 DEFR OFF5Ah CCM0 DEFR OFF5Ah CCM0 DEFR OFF5Ah CCON DEFR OFF5Ah T5CON DEFR OFF4Ah T5CON DEFR OFF4Ah T3CON DEFR OFF4Ah PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF32h PWMCON	SSCTIC	DEFR	0FF72h
SOTIC DEFR OFF6Ch CRIC DEFR OFF6Ch CRIC DEFR OFF6Ch TGIC DEFR OFF6Ch T5IC DEFR OFF6Ch T4IC DEFR OFF6Ch T3IC DEFR OFF6Ch T2IC DEFR OFF5Ch CCM3 DEFR OFF56h CCM1 DEFR OFF5Ch CCM0 DEFR OFF5Ch T01CON DEFR OFF48h T5CON DEFR OFF46h T4CON DEFR OFF42h T2CON DEFR OFF42h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF32h CCM6 DEFR OFF28h CCM5 DEFR OFF24h	SOEIC	DEFR	0FF70h
CRIC DEFR OFF6Ah T6IC DEFR OFF6Ah T5IC DEFR OFF6Ah T4IC DEFR OFF6Ah T3IC DEFR OFF6Ah T2IC DEFR OFF6Ah CCM3 DEFR OFF6Ah CCM1 DEFR OFF5Ah CCM0 DEFR OFF5Ah CCM0 DEFR OFF5Ah T0LCON DEFR OFF5Ah T6CON DEFR OFF5Ah T3CON DEFR OFF4Ah T3CON DEFR OFF32h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF32h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 </td <td>SORIC</td> <td>DEFR</td> <td>0FF6Eh</td>	SORIC	DEFR	0FF6Eh
T6IC DEFR OFF68h T5IC DEFR OFF66h T4IC DEFR OFF64h T3IC DEFR OFF60h T2IC DEFR OFF60h CCM3 DEFR OFF50h CCM2 DEFR OFF56h CCM1 DEFR OFF50h T01CON DEFR OFF50h T6CON DEFR OFF48h T5CON DEFR OFF44h T3CON DEFR OFF46h T4CON DEFR OFF40h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF32h CCM6 DEFR OFF28h CCM5 DEFR OFF24h	SOTIC	DEFR	0FF6Ch
T51C DEFR OFF66h T41C DEFR OFF64h T31C DEFR OFF62h T21C DEFR OFF60h CCM3 DEFR OFF58h CCM2 DEFR OFF56h CCM1 DEFR OFF50h T01CON DEFR OFF50h T6CON DEFR OFF40h T5CON DEFR OFF46h T4CON DEFR OFF46h T4CON DEFR OFF42h T3CON DEFR OFF42h T2CON DEFR OFF32h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF32h PWMCON0 DEFR OFF32h CCM6 DEFR OFF26h CCM5 DEFR OFF24h	CRIC	DEFR	OFF6Ah
T4IC DEFR OFF64h T3IC DEFR OFF62h T2IC DEFR OFF60h CCM3 DEFR OFF58h CCM2 DEFR OFF56h CCM1 DEFR OFF52h T01CON DEFR OFF48h T5CON DEFR OFF46h T4CON DEFR OFF42h T2CON DEFR OFF40h PWMCON1 DEFR OFF30h CCM7 DEFR OFF30h CCM6 DEFR OFF32h	TGIC	DEFR	0FF68h
T3IC DEFR OFF62h T2IC DEFR OFF60h CCM3 DEFR OFF58h CCM2 DEFR OFF56h CCM1 DEFR OFF52h T01CON DEFR OFF50h T6CON DEFR OFF50h T6CON DEFR OFF48h T5CON DEFR OFF42h T3CON DEFR OFF42h T2CON DEFR OFF42h T2CON DEFR OFF42h T2CON DEFR OFF32h PWMCON1 DEFR OFF32h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h	T5IC	DEFR	0FF66h
T2IC DEFR OFF60h CCM3 DEFR OFF58h CCM2 DEFR OFF56h CCM1 DEFR OFF52h CCM0 DEFR OFF52h T01CON DEFR OFF50h T6CON DEFR OFF48h T5CON DEFR OFF46h T4CON DEFR OFF42h T2CON DEFR OFF42h T2CON DEFR OFF42h T2CON DEFR OFF32h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF32h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF26h		DEFR	0FF64h
CCM3 DEFR OFF58h CCM2 DEFR OFF56h CCM1 DEFR OFF56h CCM0 DEFR OFF50h T01CON DEFR OFF50h T6CON DEFR OFF48h T5CON DEFR OFF46h T4CON DEFR OFF46h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h		DEFR	0FF62h
CCM2 DEFR OFF56h CCM1 DEFR OFF54h CCM0 DEFR OFF52h T01CON DEFR OFF32h T6CON DEFR OFF48h T5CON DEFR OFF46h T4CON DEFR OFF46h T4CON DEFR OFF42h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF32h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h		DEFR	
CCM1 DEFR OFF54h CCM0 DEFR OFF52h T01CON DEFR OFF50h T6CON DEFR OFF48h T5CON DEFR OFF44h T3CON DEFR OFF44h T3CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
CCM0 DEFR OFF52h T01CON DEFR OFF50h T6CON DEFR OFF48h T5CON DEFR OFF46h T4CON DEFR OFF42h T3CON DEFR OFF42h T2CON DEFR OFF42h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
T01CON DEFR OFF50h T6CON DEFR OFF48h T5CON DEFR OFF46h T4CON DEFR OFF46h T3CON DEFR OFF42h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
T6CON DEFR OFF48h T5CON DEFR OFF46h T4CON DEFR OFF46h T3CON DEFR OFF42h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
T5CON DEFR OFF46h T4CON DEFR OFF44h T3CON DEFR OFF42h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
T4CON DEFR OFF44h T3CON DEFR OFF42h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
T3CON DEFR OFF42h T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
T2CON DEFR OFF40h PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
PWMCON1 DEFR OFF32h PWMCON0 DEFR OFF30h CCM7 DEFR OFF28h CCM6 DEFR OFF26h CCM5 DEFR OFF24h			
PWMCON0 DEFR 0FF30h CCM7 DEFR 0FF28h CCM6 DEFR 0FF26h CCM5 DEFR 0FF24h			
CCM7 DEFR 0FF28h CCM6 DEFR 0FF26h CCM5 DEFR 0FF24h			
CCM6 DEFR 0FF26h CCM5 DEFR 0FF24h			
CCM5 DEFR 0FF24h			
DEFR OFF22II			
		DEFIN	0112211

99/05/08 23:03:14			
T78CON	DEFR	0FF20h	
P1H	DEFR	0FF06h	
P1L	DEFR	0FF04h	
POH POL	DEFR DEFR	0FF02h 0FF00h	
PECC7	DEFR	OFECEh	
PECC6	DEFR	0FECCh	
PECC5	DEFR	OFECAh	
PECC4	DEFR	0FEC8h	
PECC3	DEFR	0FEC6h	
PECC2 PECC1	DEFR	0FEC4h 0FEC2h	
PECCI	DEFR DEFR	0FEC2h 0FEC0h	
SRCPO	DEFA	0FCE0h	
DSTP0	DEFA	0FCE2h	
SRCP1	DEFA	0FCE4h	
DSTP1	DEFA	0FCE6h	
SRCP2 DSTP2	DEFA DEFA	0FCE8h 0FCEAh	
SRCP3	DEFA	OFCECh	
DSTP3	DEFA	OFCEEh	
SRCP4	DEFA	0FCF0h	
DSTP4	DEFA	0FCF2h	
SRCP5	DEFA	0FCF4h	
DSTP5 SRCP6	DEFA DEFA	0FCF6h 0FCF8h	
DSTP6	DEFA	OFCFAh	
SRCP7	DEFA	0FCFCh	
DSTP7	DEFA	OFCFEh	
S0BG S0RBUF	DEFR DEFR	0FEB4h	
SOTBUF	DEFR	OFEB2h, r OFEB0h, w	
WDT	DEFR	OFEAEh, r	
ADDAT	DEFR	OFEA0h	
CC15	DEFR	0FE9Eh	
CC14 CC13	DEFR DEFR	0FE9Ch 0FE9Ah	
CC12	DEFR	0FE98h	
CC11	DEFR	0FE96h	
CC10	DEFR	0FE94h	
CC9	DEFR	0FE92h	
CC8 CC7	DEFR DEFR	0FE90h 0FE8Eh	
CC6	DEFR	0FE8Ch	
CC5	DEFR	0FE8Ah	
CC4	DEFR	0FE88h	
CC3 CC2	DEFR	0FE86h	
CC1	DEFR DEFR	0FE84h 0FE82h	
CCO	DEFR	0FE80h	
CC31	DEFR	0FE7Eh	
CC30	DEFR	0FE7Ch	
CC29	DEFR	0FE7Ah	
CC28 CC27	DEFR DEFR	0FE78h 0FE76h	
CC26	DEFR	0FE74h	
CC25	DEFR	0FE72h	
CC24	DEFR	0FE70h	
CC23 CC22	DEFR	OFE6Eh	
CC21	DEFR DEFR	0FE6Ch 0FE6Ah	
CC20	DEFR	0FE68h	
CC19	DEFR	0FE66h	
CC18	DEFR	0FE64h	
CC17	DEFR	0FE62h	

CC16	DEFR	0FE60h
TIREL	DEFR	0FE56h
TOREL	DEFR	0FE54h
TI		
TO	DEFR	OFE52h
	DEFR	0FE50h
CAPREL	DEFR	OFE4Ah
тб	DEFR	0FE48h
т5	DEFR	0FE46h
т4	DEFR	OFE44h
т3	DEFR	OFE42h
т2	DEFR	OFE40h
PW3	DEFR	OFE36h
PW2	DEFR	0FE34h
PW1	DEFR	0FE32h
PWO		
PWO	DEFR	0FE30h
; Extended sfr	area	
,	ui cu	
ODP8	DEFR	0F1D6h
ODP7	DEFR	0F1D2h
ODP6	DEFR	0F1CEh
ODP3	DEFR	0F1C6h
PICON	DEFR	0F1C4h
ODP2	DEFR	0F1C2h
EXICON	DEFR	0F1C2h
SOTBIC	DEFR	0F19Ch
XP3IC	DEFR	0F19Eh
XP2IC	DEFR	0F196h
XP1IC	DEFR	0F18Eh
XPOIC	DEFR	0F186h
PWMIC	DEFR	0F17Eh
T8IC	DEFR	0F17Ch
T7IC	DEFR	0F17Ah
CC31IC	DEFR	0F194h
CC30IC	DEFR	0F18Ch
CC29IC	DEFR	0F184h
CC28IC	DEFR	0F178h
CC27IC	DEFR	0F176h
CC26IC	DEFR	0F174h
CC25IC	DEFR	0F172h
CC24IC	DEFR	0F170h
CC23IC	DEFR	0F16Eh
CC22IC	DEFR	0F16Ch
CC21IC	DEFR	0F16Ah
CC20IC	DEFR	0F168h
CC19IC	DEFR	0F166h
CC18IC	DEFR	0F164h
CC17IC	DEFR	0F162h
CC16IC	DEFR	0F160h
RPOH	DEFR	0F108h
DP1H	DEFR	0F106h
DP1L	DEFR	0F104h
DPOH	DEFR	0F102h
DPOL	DEFR	0F100h
SSCBR	DEFR	OFOB4h
SSCRB	DEFR	
		0F0B2h
SSCTB	DEFR	OFOBOh
ADDAT2	DEFR	OFOAOh
T8REL	DEFR	0F056h
T7REL	DEFR	0F054h
Т8	DEFR	0F052h
т7	DEFR	0F050h
PP3	DEFR	0F03Eh
PP2	DEFR	0F03Ch
PP1	DEFR	0F03Ah
		0 × 0 0 1 111

reg167b.def

2

99/05/08 23:03:14

	,	

		e a	1		
PPO	DEFR	0F038h	AN13	DEFB	DE 12
PT3		0F036h			P5.13
	DEFR		AN14	DEFB	P5.14
PT2	DEFR	0F034h	AN15	DEFB	P5.15
PT1	DEFR	0F032h	TGEUD	LIT	'AN10'
PTO	DEFR	0F030h	T5EUD	LIT	'AN11'
			TGIN	LIT	'AN12'
; Bit names					
			T5IN	LIT	'AN13'
CCOIO	DEFB	P2.0	T4EUD	LIT	'AN14'
CC1IO	DEFB	P2.1	T2EUD	LIT	'AN15'
CC210	DEFB	P2.2			
CC310	DEFB	P2.3	POUTO	DEFB	P7.0
CC4IO	DEFB	P2.4			
			POUT1	DEFB	P7.1
CC510	DEFB	P2.5	POUT2	DEFB	P7.2
CC6IO	DEFB	P2.6	POUT3	DEFB	P7.3
CC7IO	DEFB	P2.7	CC28I0	DEFB	P7.4
CC8IO	DEFB	P2.8	CC29I0	DEFB	P7.5
CC910	DEFB	P2.9		DEFB	
CC10IO			CC30IO		P7.6
	DEFB	P2.10	CC31I0	DEFB	P7.7
CC11IO	DEFB	P2.11			
CC12IO	DEFB	P2.12	CC16IO	DEFB	P8.0
CC13IO	DEFB	P2.13	CC17I0	DEFB	P8.1
CC14I0	DEFB	P2.14	CC18I0	DEFB	P8.2
CC15I0	DEFB	P2.15	CC19I0	DEFB	P8.3
EXOIN	LIT	(CC0IO)	CC2010	DEFB	P8.4
EX1IN	LIT	'CC1IO'	CC21I0	DEFB	P8.5
EX2IN	LIT	'CC210'	CC2210	DEFB	P8.6
EX3IN	LIT	'CC3I0'	CC2310	DEFB	
2113 111	DII	66510	CC2310	DEFB	P8.7
morn					
TOIN	DEFB	P3.0			
TGOUT	DEFB	P3.1	TOM	DEFB	T01CON.3
CAPIN	DEFB	P3.2	TOR	DEFB	T01CON.6
TJOUT	DEFB	P3.3	T1M	DEFB	T01CON.11
T3EUD	DEFB	P3.4	T1R	DEFB	T01CON.14
T2IN	DEFB	P3.7			
			T7M	DEFB	T78CON.3
T3IN	DEFB	P3.6	T7R	DEFB	T78CON.6
T4IN	DEFB	P3.5	T8M	DEFB	T78CON.11
SSDI	DEFB	P3.8	T8R	DEFB	T78CON.14
SSDO	DEFB	P3.9			
TXD0	DEFB	P3.10	ACC0	DEFB	CCM0.3
RXD0	DEFB	P3.11			
			ACC1	DEFB	CCM0.7
SSCLK	DEFB	P3.13	ACC2	DEFB	CCM0.11
CLKOUT	DEFB	P3.15	ACC3	DEFB	CCM0.15
A16	DEFB	P4.0	ACC4	DEFB	CCM1.3
A17	DEFB	P4.1	ACC5	DEFB	CCM1.7
A18	DEFB	P4.2	ACC6	DEFB	CCM1.11
A19	DEFB	P4.3	ACC7	DEFB	
A20	DEFB		ACC	DEFB	CCM1.15
		P4.4			
A21	DEFB	P4.5	ACC8	DEFB	CCM2.3
A22	DEFB	P4.6	ACC9	DEFB	CCM2.7
A23	DEFB	P4.7	ACC10	DEFB	CCM2.11
			ACC11	DEFB	CCM2.15
AN0	DEFB	P5.0			
AN1	DEFB	P5.1	ACC12	DEFB	CCM3.3
AN2	DEFB	P5.2			
			ACC13	DEFB	CCM3.7
AN3	DEFB	P5.3	ACC14	DEFB	CCM3.11
AN4	DEFB	P5.4	ACC15	DEFB	CCM3.15
AN5	DEFB	P5.5			
AN6	DEFB	P5.6	ACC16	DEFB	CCM4.3
AN7	DEFB	P5.7			
			ACC17	DEFB	CCM4.7
AN8	DEFB	P5.8	ACC18	DEFB	CCM4.11
AN9	DEFB	P5.9	ACC19	DEFB	CCM4.15
AN10	DEFB	P5.10			
AN11	DEFB	P5.11	ACC20	DEFB	CCM5.3
AN12	DEFB	P5.12	ACC21	DEFB	CCM5.7
1000 (1007 (170)			ACC 41	DEFD	CCHJ. /

23:03:14		reg167	b.def	
ACC22	DEFB	CCM5.11	SSCRIE	DEFB
ACC23	DEFB	CCM5.15	SSCRIR	DEFB
			SSCEIE	DEFB
ACC24 ACC25	DEFB DEFB	CCM6.3 CCM6.7	SSCEIR	DEFB
ACC26	DEFB	CCM6.11	SSCTE	LIT
ACC27	DEFB	CCM6.15	SSCRE	LIT
	2212		SSCPE SSCBE	LIT LIT
ACC28	DEFB	CCM7.3	000222	BII
ACC29	DEFB	CCM7.7		
ACC30	DEFB	CCM7.11	CCOIE	DEFB
ACC31	DEFB	CCM7.15	CCOIR	DEFB
T2R	DEFB	T2CON.6	CC1IE	DEFB
T2UD	DEFB	T2CON. 7	CC1IR CC2IE	DEFB DEFB
T2UDE	DEFB	T2CON.8	CC2IR	DEFB
			CC3IE	DEFB
T3R	DEFB	T3CON.6	CC3IR	DEFB
T3UD	DEFB	T3CON.7	CC4IE	DEFB
T3UDE T3OE	DEFB	T3CON . 8	CC4IR	DEFB
TJOTL	DEFB DEFB	T3CON.9 T3CON.10	CC5IE	DEFB
15011	DELD	15CON. 10	CC5IR CC6IE	DEFB
T4R	DEFB	T4CON.6	CC6IR	DEFB DEFB
T4UD	DEFB	T4CON.7	CC7IE	DEFB
T4UDE	DEFB	T4CON.8	CC7IR	DEFB
			CC8IE	DEFB
T5R	DEFB	T5CON. 6	CC8IR	DEFB
T5UD	DEFB	T5CON.7	CC9IE	DEFB
T5UDE T5CLR	DEFB DEFB	T5CON.8 T5CON.14	CC9IR	DEFB
T5SC	DEFB	T5CON.14	CC10IE	DEFB
1000		15C0N.15	CC10IR CC11IE	DEFB DEFB
T6R	DEFB	T6CON.6	CC11IR	DEFB
T6UD	DEFB	T6CON.7	CC12IE	DEFB
T6UDE	DEFB	T6CON.8	CC12IR	DEFB
T60E T60TL	DEFB	T6CON . 9	CC13IE	DEFB
T6SR	DEFB DEFB	T6CON.10 T6CON.15	CC13IR	DEFB
TODIC	DEFD	18CON.15	CC14IE CC14IR	DEFB
T2IE	DEFB	T2IC.6	CC15IE	DEFB DEFB
T2IR	DEFB	T2IC.7	CC15IR	DEFB
TJIE	DEFB	T3IC.6	CC16IE	DEFB
T3IR	DEFB	T3IC.7	CC16IR	DEFB
T4IE T4IR	DEFB DEFB	T4IC.6 T4IC.7	CC17IE	DEFB
T5IE	DEFB	T5IC.6	CC17IR	DEFB
T5IR	DEFB	T5IC.7	CC18IE CC18IR	DEFB
T6IE	DEFB	T6IC.6	CC19IE	DEFB
T6IR	DEFB	TGIC.7	CC19IR	DEFB
0575			CC20IE	DEFB
CRIE CRIR	DEFB DEFB	CRIC.6 CRIC.7	CC20IR	DEFB
CRIK	DEFB	CRIC. /	CC21IE CC21IR	DEFB
SOTIE	DEFB	SOTIC.6	CC221E	DEFB DEFB
SOTIR	DEFB	SOTIC.7	CC22IE CC22IR	DEFB
SORIE	DEFB	SORIC.6	CC23IE	DEFB
SORIR	DEFB	SORIC.7	CC23IR	DEFB
SOEIE	DEFB	SOEIC.6	CC24IE	DEFB
SOEIR SOTBIE	DEFB DEFB	SOEIC.7	CC24IR	DEFB
SOTBIE	DEFB	SOTBIC.6 SOTBIC.7	CC25IE	DEFB
	2010		CC25IR CC26IE	DEFB DEFB
SSCTIE	DEFB	SSCTIC.6	CC26IE CC26IR	DEFB
SSCTIR	DEFB	SSCTIC.7	CC27IE	DEFB

99/05/08

SSCRIC.6 SSCRIC.7 SSCEIC.6 SSCEIC.7 'SSCTEN' 'SSCREN' 'SSCPEN'

CCOIC.6 CCOIC.7 CC1IC.6 CC1IC.7 CC2IC.6 CC2IC.7 CC3IC.6 CC3IC.7 CC4IC.6 CC4IC.7 CC5IC.6 CC5IC.7 CC6IC.6 CC6IC.7 CC7IC.6 CC7IC.7 CC8IC.6 CC8IC.7 CC9IC.6 CC9IC.7 CC10IC.6 CC10IC.7 CC11IC.6 CC11IC.7 CC12IC.6 CC12IC.7 CC13IC.6 CC13IC.7 CC14IC.6 CC14IC.7 CC15IC.6 CC15IC.7 CC16IC.6 CC16IC.7 CC17IC.6 CC17IC.7 CC18IC.6 CC18IC.7 CC19IC.6 CC19IC.7 CC20IC.6 CC20IC.7 CC21IC.6 CC21IC.7 CC22IC.6 CC22IC.7

CC23IC.6

CC23IC.7

CC24IC.6

CC24IC.7

CC25IC.6

CC25IC.7 CC26IC.6

CC26IC.7

CC27IC.6

99/05/08 23:03:14		
CC27IR	DEFB	CC27IC.7
CC28IE	DEFB	CC28IC.6
CC28IR	DEFB	CC28IC.7
CC29IE	DEFB	CC29IC.6
CC29IR	DEFB	CC29IC.7
CC30IE	DEFB	CC30IC.6
CC30IR	DEFB	CC30IC.7
CC31IE	DEFB	CC31IC.6
CC31IR	DEFB	CC31IC.7
ADCIE	DEFB	ADCIC.6
ADCIR	DEFB	ADCIC.7
ADEIE	DEFB	ADEIC.6
ADEIR	DEFB	ADEIC.7
TOIE	DEFB	TOIC.6
TOIR	DEFB	TOIC.7
TIIE	DEFB	T1IC.6
T1IR	DEFB	T11C.7
T7IE	DEFB	T7IC.6
T7IR	DEFB	T7IC.7
T8IE	DEFB	T8IC.6
T8IR	DEFB	T8IC.7
ADST	DEFB	ADCON.7
ADBSY	DEFB	ADCON.8
ADWR	DEFB	ADCON.9
ADCIN	DEFB	ADCON.10
ADCRQ	DEFB	ADCON.11
ILLBUS	DEFB	TFR.0
ILLINA	DEFB	TFR.1
ILLOPA	DEFB	TFR.2
PRTFLT	DEFB	TFR.3
	DEFB	TFR. 7
UNDOPC		
STKUF	DEFB	TFR.13
STKOF	DEFB	TFR.14
NMI	DEFB	TFR.15
WDTIN	DEFB	WDTCON.0
WDTR	DEFB	WDTCON.1
SOSTP	DEFB	SOCON.3
SOREN	DEFB	SOCON.4
SOPEN	DEFB	SOCON.5
SOFEN	DEFB	SOCON.6
SOOEN	DEFB	SOCON.7
SOPE	DEFB	SOCON.8
SOFE	DEFB	SOCON.9
SOOE	DEFB	SOCON.10
SOODD	DEFB	SOCON.12
SOBRS	DEFB	SOCON.13
SOLB	DEFB	SOCON.14
SOR	DEFB	S0CON.15
SSCHB	DEFB	SSCCON.4
SSCPH	DEFB	SSCCON.5
SSCPO	DEFB	SSCCON.6
SSCTEN	DEFB	SSCCON.8
SSCREN	DEFB	SSCCON.9
SSCPEN	DEFB	SSCCON.10
SSCBEN		SSCCON.10
	DEFB	
SSCBSY	DEFB	SSCCON.12
SSCMS	DEFB	SSCCON.14
SSCEN	DEFB	SSCCON.15

PTR0	DEFB	PWMCON0.0
PTR1	DEFB	PWMCON0.1
PTR2	DEFB	PWMCON0.2
PTR3	DEFB	PWMCON0.3
PTIO	DEFB	PWMCON0.4
PTI1	DEFB	PWMCON0.5
PTI2	DEFB	PWMCON0.6
PTI3	DEFB	PWMCON0.7
PIEO	DEFB	PWMCON0.8
PIE1	DEFB	PWMCON0.9
PIE2	DEFB	PWMCON0.10
PIE3	DEFB	PWMCON0.11
PIR0	DEFB	PWMCON0.12
PIR1	DEFB	PWMCON0.13
PIR2	DEFB	PWMCON0.14
PIR3	DEFB	PWMCON0.15
PENO	DEFB	PWMCON1.0
PEN1	DEFB	PWMCON1.1
PEN2	DEFB	PWMCON1.2
PEN3	DEFB	PWMCON1.3
PMO	DEFB	PWMCON1.4
PM1	DEFB	PWMCON1.5
PM2	DEFB	PWMCON1.6
PM3	DEFB	PWMCON1.7
PB01	DEFB	PWMCON1.12
PS2	DEFB	PWMCON1.14
PS3	DEFB	PWMCON1.15
PWMIE	DEFB	PWMIC.6
PWMIR	DEFB	PWMIC.7
XP3IE	DEFB	XP3IC.6
XP3IR	DEFB	XP3IC.7
XP2IE	DEFB	XP2IC.6
XP2IR	DEFB	XP2IC.7
XP1IE	DEFB	XP1IC.6
XP1IR	DEFB	XP1IC.7
XPOIE	DEFB	XP0IC.6
XPOIR	DEFB	XPOIC.7

B.4 14V Bus CAN Node 3

On the next page starts the code for the 14V bus CAN node 3. The files for the node are as follows.

- 1. comp312.bat
- 2. main312.asm
- 3. cnmod312.asm
- $4. \ canmo312.asm$
- 5. cnint312.asm
- 6. atod312.asm
- 7. tmrs312.asm
- 8. linker.lnv
- 9. Reg167b.def



al66 main312.asm al66 cnmod312.asm al66 cnint312.asm al66 cnint312.asm al66 atod312.asm al66 tmrs312.asm al66 tmrs312.asm ll66 LINK main312.obj cnmod312.obj canmo312.obj cnint312.obj atod312.obj tmrs312.obj TO locatein.lno ll66 @linker.lnv

ihex166 -i16 locate.out -o main312.hex



main312.asm



;; Initialize CAN Bus SSEGMENTED ; Call the CAN initialization function CALL canin **\$EXTEND** ;; End of CAN Bus Initialization **\$EXTSFR** : CAN USE ALL internal RAM for Stack SEXTSSK meto: \$EXTMEM ; just loop here waiting NOP SNOMOD166 NOP \$STDNAMES(reg167b.def) JMP meto \$SYMBOLS RET ; return main ENDP NAME main mainseg ENDS ; define a common register area of 16 register RBANK1 COMREG R0-R15 ; codesegment that contains reset int pointer startupsec SECTION CODE ; default stack size of 256 Words SSKDEF 4 sysreset PROC TASK INTNO=0H ; reset interrupt number is zero at Oh ; forces next instruction to be located at 0h ORG 000H ASSUME DPP3:SYSTEM ; installs a pointer to the startup routine JMP start ; return from interrupt RETI : Can function EXTERN canin: FAR sysreset ENDP ; external atod initialization EXTERN atod initialize:FAR startupsec ENDS EXTERN atod timer_initialize:FAR END mainseg SECTION CODE main PROC FAR ; disable the watchdog timer start: DISWDT ; Globally Enable Interrupts both global BSET IEN ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh ; end initialization EINIT ;; End of external memory bus initialization ;; Initialize the Data Page pointers for this section ; make DPP3 point to system MOV DPP3, #03h ;; End of Data Page Pointer Initialization ;; Make the direction of Port 2 to output MOV DP2, ONES ;; Make sure Port 2 is in push/pull mode MOV ODP2, ONES ;; Initialize The Stack ;; The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #OFBFFh the highest ;; byte that the stack pointers can point to is #OFBFEh MOV STKUN, #0FBFEh; Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP, #0FBFEh ; Set the Stack Pointer ;; End of Stack Initialization ;; Initialize the Analog to Digital Converter CALL atod_initialize; atod ;; End of A/D initialization ;; Initialize A/D timer CALL atod_timer_initialize; timers ;; End of A/D timer initialization

99/05/09 11:00:32

cnmod312.asm



\$SEGMENTED RET \$EXTEND canin \$EXTSFR setall PROC FAR \$EXTMEM ;; by using a counter it counts up to 15 and initializes all of the Mess objects along the way.	invalid
\$STDNAMES(reg16/b.der) \$SYMBOLS PUSH R4	e message
NAME canmod PUSH R5 AND R5, ZEROS	
RBANK1 COMREG R0-R15 ; define a common register area of 16 registers OR R5, #01h ; Set counter to 1 for first MO GLOBAL canin ; The function must be declared Global at the AND R2, ZEROS OR R2, #0EF10h ; Set pointer to MO1 , beginning of the module AND R4, ZEROS OR R4 #5555h : Set R4 to make MObs invalid	
EXTERN canmodig:FAR ; configures specific Message objects	
ASSUME DPP3:SYSTEM nextreg:MOV [R2],R4 ; make all message objects invalid ADD R2,#10h CMPI1 R5,#0Fh	
canfunc SECTION CODE ; codesegment that contains reset int pointer JMPA CC_NZ, nextreg ; canin PROC FAR PUSH R0 PUSH R1 POP R2 RET	
;; set all of the CAN control registerssetall ENDPAND C1CSR,ZEROS; set control register to zerocanfunc ENDSMOV R1, #0043h; Set IE and INIT bitscanfunc ENDSOR C1CSR,R1; set control register to R1's valueEND	
AND C1BTR, ZEROS ; set Bit timing register to zero MOV R1, #03447h ; set for 125k operation OR C1BTR, R1 ; set Bit timing register parameters	
AND CIGMS, ZEROS ; set Global Mask short register to zero MOV R1, #OFFFFh ; EOFF is what DAVE initialize OR CIGMS, Rl ; set GMS	
AND ClUGML, ZEROS ; set Upper global mask long to zero MOV R1, #0FFFFh OR ClUGML, R1	
MOV R1, #0F8FFh AND C1LGML, ZEROS OR C1LGML, R1 ; lower global mask	
AND C1UMLM, ZEROS OR C1UMLM, R1 ; upper mask of last register AND C1LMLM, ZEROS OR C1LMLM, R1 ; lower mask of last register	
CALL setall ; sets all of the CAN registers to off	
CALL canmocfg ; Configures specific Message Objects	
<pre>;; Setup CAN interrupt and Initialize CAN module EXTR #4 AND XPOIC, ZEROS ; configure CAN interrupt control Register AND R0,ZEROS OR R0,#0073h ; enable interrupt, level is 10 group is 2 OR XPOIC,R0 ; Configure CAN interrupt Control Register AND R1, ZEROS OR R1, #00041h ; crashes if I clear the CPU access to the BTR XOR CICSR, R1 ; end initialize CAN interrupt POP R1 POP R0</pre>	

99/05/11 16:50:38 canmo3	12.asm
\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS NAME canmo RBANK1 COMREG R0-R15 ; declare bank of 16 global registers	MOV [R2],R1; set M03's Control registerADD R2,#2h; point to Upper Arbitration registerAND R3, ZEROS; set R6 to zeroOR R3, #0E002h; The number is the Message ID for Message Object 3MOV [R2],R3; message id = 0ADD R2, #2h; Point to the Lower Arbitration RegisterMOV [R2], ZEROS; standard Message object so lowerarb = 0hAND R1, ZEROS; put 000h into first data byte and set to receiveMOV MCD_M3,R1; Databyte(0) = 0 and Set to receive and 3 bytes o
GLOBAL canmocfg	f data MOV DATA_M3, ZEROS ; Fill the Data of the MO with Zeros
<pre>can_module SECTION CODE ASSUME DPP3:SYSTEM canmocfg PROC FAR PUSH R1 PUSH R2 PUSH R3 ;; Now set specific CAN control Registers ;; initialize message object 1 ;; initializing this object to be invalid does or removing the code until ;; the comment "Setup CAN interrupt and Initialize" does ;; nothing to prevent the occurrance of the interrupt for the CAN system MOV R2, #MCR_M1 ; start of Message Object 1 AND R1, ZEROS OR R1, #5599h ; Generate a Receive Interrupt if this message object ac tivates</pre>	<pre>;; Initialize Message Object 4 MOV R2, #MCR_M4 ; start of Message Object 4 AND R1, ZEROS OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register ADD R2, #2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero OR R3, #0002h ; The number is the Message ID for Message Object 4 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0038h ; put 0AAh into first data byte and set to receive MOV MCD_M4,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o MOV DATA_M4, ZEROS ; fill the data of the M0 with ZEROS</pre>
<pre>MOV [R2],R1 ; set MO1's Control register ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R3 to OR R3, #0C001h ; message id for message object 1 MOV [R2],R3 ; message id = #0003h ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0030h ; put 0AAh into first data byte and set to receive MOV MCD_M1,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes of data MOV DATA_M1, ZEROS ; fill the Data of the MO with Zeros ;; Initialize Message Object 2 MOV R2, #MCR_M2 ; start of Message Object 2 AND R1, ZEROS OR R1, #5599h ; RECEIVE INTERRUPT enabled MOV [R2],R1 ; set MO2's Control register ADD R2, #2h ; point to Upper Arbitration register</pre>	<pre>;; Initialize Message Object 5 MOV R2, #MCR_M5 ; start of Message Object 5 AND R1, ZEROS OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero OR R3, #00013h ; The number is the Message ID for Message Object 5 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = Oh AND R1, ZEROS OR R1, #0038h ; put OAAh into first data byte and set to receive MOV MCD_M5,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o f data MOV DATA_M5, ZEROS ; fill the data of the M0 with ZEROS</pre>
AND R3, ZEROS ; set R6 to zero OR R3, #0E001h ; The number is the Message ID for Message Object 2 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0030h ; put 000h into first data byte and set to receive MOV MCD_M2,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes of da ta MOV DATA_M2, ZEROS ; Fill the Data of the MO with Zeros ;; Initialize Message Object 3 MOV R2, #MCR_M3 ; start of Message Object 3 AND R1, ZEROS OR R1, #5595h ; Generate a receive interrupt if this message object ac tivates	OR R3, #00014h ; The number is the Message in for Message object of Move [R2], R3 MOV [R2], R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS ; put 0AAh into first data byte and set to receive



ta

ta

ta

END

canmo312.asm



; fill the data of the MO with ZEROS MOV DATA M6, ZEROS ;; Initialize Message Object 7 ; start of Message Object 7 MOV R2, #MCR_M7 AND R1, ZEROS OR R1, #5599h ; set MO7's Control register MOV [R2], R1 ; point to Upper Arbitration register ADD R2,#2h ; set R6 to zero AND R3, ZEROS ; The number is the Message ID for Message Object 7 OR R3, #00022h ; message id = 0 MOV [R2], R3 ; Point to the Lower Arbitration Register ADD R2, #2h ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; put OAAh into first data byte and set to receive OR R1, #0030h ; Databyte(0) = 0 and Set to receive and 3 bytes of da MOV MCD_M7,R1 ; fill the data of the MO with ZEROS MOV DATA_M7, ZEROS ;; Initialize Message Object 8 ; start of Message Object 8 MOV R2, #MCR_M8 AND R1, ZEROS OR R1, #5595h ; set MO8's Control register MOV [R2], R1 ; point to Upper Arbitration register ADD R2,#2h ; set R6 to zero AND R3, ZEROS ; The number is the Message ID for Message Object 8 OR R3, #00023h ; message id = 0 MOV [R2], R3 ; Point to the Lower Arbitration Register ADD R2, #2h ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; put OAAh into first data byte and set to receive OR R1, #0038h ; Databyte(0) = 0 and Set to receive and 3 bytes of da MOV MCD_M8, R1 ; fill the data of the MO with ZEROS MOV DATA_M8, ZEROS ;; Initialize Message Object 9 MOV R2, #MCR_M9 : start of Message Object 9 AND R1, ZEROS OR R1, #5595h ; set MO9's Control register MOV [R2], R1 ; point to Upper Arbitration register ADD R2, #2h ; set R6 to zero AND R3, ZEROS ; The number is the Message ID for Message Object 9 OR R3, #00024h ; message id = 0 MOV [R2], R3 ; Point to the Lower Arbitration Register ADD R2, #2h ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; put OAAh into first data byte and set to receive OR R1, #0038h ; Databyte(0) = 0 and Set to receive and 3 bytes of da MOV MCD_M9,R1 ; fill the data of the MO with ZEROS MOV DATA M9, ZEROS POP R3 POP R2 POP R1 RET canmocfg ENDP can_module ENDS



\$SEGMENTED SEXTEND SEXTSFR SEXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS NAME canint ; declare bank of 16 global registers RBANK1 COMREG R0-R15 ASSUME DPP3:SYSTEM can_interrupts SECTION CODE can_receive_interrupt PROC TASK INTNO=040h ORG 0100h CALL can_receive_interrupt_handler RETI can_receive_interrupt ENDP can_receive_interrupt_handler PROC FAR PUSH RO PUSH R1 PUSH R2 ; Read the CAN interrupt ID buffer MOVB RLO, INTID ; See if the interrupt came from M01 CMPB RLO, #03h JMP cc_Z, message_one_interrupt; if interrupt from M01 handle ; See if the interrupt came from M07 CMPB RLO, #09h JMP cc_Z, message_seven_interrupt MOV R1, #05555h MOV R2, #05599h MOV MCR_M2, R1 MOV RO, DATA_M2 MOV MCR_M2, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch MOV R2, MCR_M6 MOV MCR_M6, R1 MOV DATA M6, R0 MOV MCR M6, R2 CMP R0, #01h JMP cc_NZ, turn_off_heated_rear_window BSET P2.1 JMP exit_function turn_off_heated_rear_window: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.1 JMP exit_function message_one_interrupt: MOV R1, #05555h MOV R2, #05599h MOV MCR_M1, R1 MOV RO, DATA_M1 MOV MCR_M1, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch



MOV R2, MCR_M5 MOV MCR_M5, R1 MOV DATA_M5, R0

MOV MCR_M5, R2 CMP R0, #01h JMP cc_NZ, turn_heater_off BSET P2.0 JMP exit_function

turn_heater_off: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.0 JMP exit_function

message_seven_interrupt: MOV R1, #05555h MOV R2, #05599h MOV MCR_M7, R1 MOV R0, DATA_M7 MOV MCR_M7, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch

MOV R2, MCR_M9 MOV MCR_M9, R1 MOV DATA_M9, R0

MOV MCR_M9, R2 CMP R0, #01h JMP cc_NZ, turn_off_bridge BSET P2.2 JMP exit_function

turn_off_bridge: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.2 JMP exit_function

exit_function: MOV R2, #0EFFFh AND C1CSR, R2 POP R2 POP R1 POP R0 RET can_receive_interrupt_handler ENDP

can_interrupts ENDS END



atod312.asm



PUSH R3 \$SEGMENTED PUSH R4 SEXTEND PUSH MDH SEXTSFR PUSH MDL ; CAN USE ALL internal RAM for Stack SEXTSSK MOV R2, ADDAT SEXTMEM ; This is so we can isolate the A/D channel from whi MOV R0, R2 \$NOMOD166 ch the data is coming \$STDNAMES(reg167b.def) MOV R3, R2 \$SYMBOLS ; No Scaling on Microcontroller MOV R4, #01h AND R0, #0F000h ; The channel information is located in the upper nibble CMP R0, #01000h ; See if the information is coming from Channel 1 of the A/ name atod D JMP cc_Z, break_loads_current ASSUME DPP3:SYSTEM CMP R0, #02000h ; See if the information is coming from Channel 2 of the A/D RBANK1 COMREG R0-R15 JMP cc Z. Voltage Bridge_current GLOBAL atod_initialize ; This bit pattern deactives MCRs MOV R0, #05555h MOV R1, MCR_M3 ; SAVE the Configuration of the MCR ;; This A/D is set up to measure the current in two different ; Kill the Message Control Register MOV MCR M3, R0 ;; loads. Because this software is to be used as part of ;; 42volt bus node 1, it uses the names of the loads that MUL R3, R4 ;; that node is supposed to control. NOP ;; The analog to digital converter uses Port 5 ; for real MOV DATA_M3, MDL ; for testing purposes MOV P2, R2 ; MOV MCR M3, R1 atod_setup SECTION CODE BSET T3R JMP exit routine atod_initialize PROC FAR ;; Initialize variables Break_loads_current: ;; This below line of code setups up the A/D converter ;; for 2 channels and single conversion. ; This bit pattern deactives MCRs MOV R0, #05555h ;; It is also set for "Wait for read mode" : SAVE the Configuration of the MCR MOV R1, MCR_M4 ;; so the converter will wait for the user program to read ; Kill the Message Control Register MOV MCR_M4, R0 ;; the buffer before processing the next channel. MOV R0, #08h ;test code MOV ADCON, #0A222h ; setup A/D control register ADD P2, R0 ;test code MUL R3, R4 ;; Set the channel to which the data should be written ;; when the first "A/D is done" interrupt occurs NOP MOV DATA_M4, MDL ; for testing purposes MOV MCR_M4, R1 ;; The below code sets up the A/D's Interrupt control register JMP exit_routine ;; The A/D is setup to have a group of 2 and a level of 10 MOV ADCIC, #006Fh Voltage_Bridge_current: RET ; This bit pattern deactives MCRs MOV R0, #05555h atod_initialize ENDP : SAVE the Configuration of the MCR MOV R1, MCR_M8 atod_setup ENDS : Kill the Message Control Register MOV MCR_M8, R0 atod_handlers SECTION CODE MUL R3, R4 atod_handler PROC TASK INTNO=028h NOP ORG 0A0H MOV DATA M4, MDL ; for testing purposes CALL atod_function MOV MCR_M4, R1 RETT JMP exit_routine atod handler ENDP atod_function PROC FAR ;; this function works by seeing if the converter is converting exit_routine: ;; for the heater_measurement. If the bit is set, then POP MDL ;; the bit gets cleared and the IP jumps to where the POP MDH ;; value in the converter is moved into the heater_current POP R4 ;; variable. POP R3 ;; otherwise the bit gets set and the value is moved into POP R2 ;; the heated_rear_window_current variable POP R1 PUSH RO POP RO PUSH R1 RET PUSH R2



atod_function ENDP atod_handlers ENDS

END





END

; These are assembler controls \$SEGMENTED SEXTEND \$EXTSFR \$EXTMEM **\$EXTINSTR** \$NOMOD166 \$STDNAMES(reg167b.def) ; Assembler controls end here \$SYMBOLS NAME timer_functions ASSUME DPP3:SYSTEM RBANK1 COMREG R0-R15 GLOBAL atod_timer_initialize atod_timer SECTION CODE atod_timer_initialize PROC FAR ; setup Core Timer T3 MOV T3CON, #0004h MOV T3IC, #002Bh ; Make the value in the counter equal to zero MOV T3, #0000h ; enable the timer interrupt BSET T3IE ; start the timer BSET T3R RET atod_timer_initialize ENDP atod_interrupt PROC TASK INTNO=023h ORG 08Ch CALL atod_timer_handler RETI atod_interrupt ENDP atod_timer_handler PROC FAR ; stop the timer BCLR T3R ; start an A/D conversion BSET ADST RET atod_timer_handler ENDP atod_timer ENDS



tmrs312.asm



LOCATE locatein.lno (GENERAL) IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (040000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO locate.out



linker.lnv



99/05/08 23:03:14

0FFD4h

0FFD2h

0FFD0h

OFFCEh

0FFCCh

OFFCAh

0FFC8h

0FFC6h

0FFC4h

0FFC2h

OFFCOh

0FFB2h

0FFB0h

OFFAEh

0FFACh

0FFA2h

OFFA0h

0FF9Eh

0FF9Ch

0FF9Ah

0FF98h 0FF96h

0FF94h

0FF92h

0FF90h

0FF8Eh

0FF8Ch

0FF8Ah

0FF88h

0FF86h

0FF84h 0FF82h

0FF80h

0FF7Eh

0FF7Ch 0FF7Ah

0FF78h

0FF76h

0FF74h

0FF72h

0FF70h 0FF6Eh

0FF6Ch

0FF6Ah

0FF68h 0FF66h

0FF64h

0FF62h

0FF60h

0FF58h 0FF56h

0FF54h

0FF52h

0FF50h

0FF48h

0FF46h

0FF44h

0FF42h

0FF40h

0FF32h

0FF30h

0FF28h

0FF26h

0FF24h

0FF22h

Contraction of the local distribution of the				1	
*************				P8	DEFR
	** @(#)reg167b.def 1.10 12/18/97			DP7	DEFR
	; ** *			P7	DEFR
tt Desigto	** Register definitions for the SAB C167				DEFR
;** Registe	A Register definitions lot the pamor and RIT pamor			P6	DEFR
;** This II	;** This file contains all SFR names and BIT names ;** This file can be supplied to rm166 and a166 (STDNAMES control)				DEFR
;** This fi	le can be	**************************************	(SIDNAMES CONCLOI)	DP4 P4	DEFR
				DP3	DEFR
TRUE	DEFB	OFF20h.0, RW		P3	DEFR
NODE142	DEFB	OFF20h.1, RW		DP2	DEFR
C1CSR	DEFA	0EF00h		P2	DEFR
INTID	DEFA	0EF02h		SSCCON	DEFR
CIBTR	DEFA	0EF04h		SOCON	DEFR
CIGMS	DEFA	0EF06h		WDTCON	DEFR
CIUGML	DEFA	0EF08h		TFR	DEFR
CILGML	DEFA	0EF0Ah		P5	DEFR
CIUMLM	DEFA	0EF0Ch		ADCON	DEFR
	DEFA	0EF0Eh		T1IC	DEFR
CILMLM				TOIC	DEFR
MCR_M1	DEFA	0EF10h		ADEIC	DEFR
MCR_M2	DEFA	0EF20h		ADCIC	DEFR
MCR_M3	DEFA	0EF30h		CC15IC	DEFR
MCR_M4	DEFA	0EF40h			DEFR
MCR_M5	DEFA	0EF50h		CC14IC	
MCR_M6	DEFA	0EF60h		CC13IC	DEFR
MCR_M7	DEFA	0EF70h		CC12IC	DEFR
MCR_M8	DEFA	0EF80h		CC11IC	DEFR
MCR_M9	DEFA	0EF90h		CC10IC	DEFR
MCR_MA	DEFA	0EFA0h		CC9IC	DEFR
MCR_MB	DEFA	0EFB0h		CC8IC	DEFR
	DEFA	0EFC0h		CC7IC	DEFR
MCR_MC		0EFD0h		CC6IC	DEFR
MCR_MD	DEFA			CC5IC	DEFR
MCR_ME	DEFA	OEFEOh		CC4IC	DEFR
MCR_MF	DEFA	0EFF0h		CC3IC	DEFR
MCD_M1	DEFA	0EF16h		CC2IC	DEFR
MCD_M2	DEFA	0EF26h			DEFR
MCD_M3	DEFA	0EF36h		CC1IC	DEFR
MCD_M4	DEFA	0EF46h		CCOIC	
MCD_M5	DEFA	0EF56h		SSCEIC	DEFR
MCD_M6	DEFA	0EF66h		SSCRIC	DEFR
MCD_M7	DEFA	0EF76h		SSCTIC	DEFR
MCD_M8	DEFA	0EF86h		SOEIC	DEFR
MCD_M9	DEFA	0EF96h		SORIC	DEFR
MCD_MA	DEFA	0EFA6h		SOTIC	DEFR
MCD_MB	DEFA	0EFB6h		CRIC	DEFR
	DEFA	0EFC6h		TGIC	DEFR
MCD_MC				T5IC	DEFR
MCD_MD	DEFA	0EFD6h		T4IC	DEFR
MCD_ME	DEFA	0EFE6h		T3IC	DEFR
DATA_M1	DEFA	0EF18h		T2IC	DEFR
DATA_M2	DEFA	0EF28h		CCM3	DEFR
DATA_M3	DEFA	0EF38h			DEFR
DATA_M4	DEFA	0EF48h		CCM2	
DATA_M5	DEFA	0EF58h		CCM1	DEFR
DATA_M6	DEFA	0EF68h		CCM0	DEFR
DATA_M7	DEFA	0EF78h		TOICON	DEFR
DATA_M8	DEFA	0EF88h		T6CON	DEFR
DATA_M9	DEFA	0EF98h		T5CON	DEFR
DATA_MA	DEFA	0EFA8h		T4CON	DEFR
DATA_MB	DEFA	0EFB8h		T3CON	DEFR
DATA_MC	DEFA	0EFC8h		T2CON	DEFR
		0EFD8h		PWMCON1	DEFR
DATA_MD	DEFA			PWMCON0	DEFR
DATA_ME	DEFA	0EFE8h		CCM7	DEFR
				CCM6	DEFR
				CCM5	DEFR
0.000	1000 1000 00 Percent	2 mm 61		CCM4	DEFR
DP8	DEFR	0FFD6h		CCI14	DEFIC
				L. C.	

2

reg167b.def

23:03:14		
T78CON	DEFR	0FF20h
P1H	DEFR	0FF06h
P1L	DEFR	0FF04h 0FF02h
POH POL	DEFR DEFR	0FF02h
PECC7	DEFR	OFECEh
PECC6	DEFR	OFECCh
PECC5	DEFR	OFECAh
PECC4	DEFR	0FEC8h 0FEC6h
PECC3 PECC2	DEFR DEFR	0FEC4h
PECC1	DEFR	0FEC2h
PECC0	DEFR	OFECOh
SRCP0	DEFA	OFCEOh
DSTP0	DEFA DEFA	0FCE2h 0FCE4h
SRCP1 DSTP1	DEFA	0FCE6h
SRCP2	DEFA	0FCE8h
DSTP2	DEFA	OFCEAh
SRCP3	DEFA	OFCECh
DSTP3	DEFA DEFA	0FCEEh 0FCF0h
SRCP4 DSTP4	DEFA	0FCF2h
SRCP5	DEFA	0FCF4h
DSTP5	DEFA	0FCF6h
SRCP6	DEFA	0FCF8h
DSTP6	DEFA	0FCFAh 0FCFCh
SRCP7 DSTP7	DEFA DEFA	OFCFEh
SOBG	DEFR	0FEB4h
SORBUF	DEFR	OFEB2h, r
SOTBUF	DEFR	OFEBOh, w
WDT	DEFR	OFEAEh, r OFEA0h
ADDAT CC15	DEFR DEFR	0FE9Eh
CC14	DEFR	0FE9Ch
CC13	DEFR	0FE9Ah
CC12	DEFR	0FE98h
CC11	DEFR DEFR	0FE96h 0FE94h
CC10 CC9	DEFR	0FE92h
CC8	DEFR	0FE90h
CC7	DEFR	0FE8Eh
CC6	DEFR	0FE8Ch
CC5	DEFR	0FE8Ah 0FE88h
CC4 CC3	DEFR	0FE86h
CC2	DEFR	0FE84h
CC1	DEFR	0FE82h
CC0	DEFR	0FE80h 0FE7Eh
CC31 CC30	DEFR DEFR	0FE7Ch
CC29	DEFR	0FE7Ah
CC28	DEFR	0FE78h
CC27	DEFR	0FE76h
CC26	DEFR	0FE74h 0FE72h
CC25 CC24	DEFR DEFR	OFE70h
CC23	DEFR	0FE6Eh
CC22	DEFR	0FE6Ch
CC21	DEFR	0FE6Ah
CC20	DEFR	0FE68h 0FE66h
CC19 CC18	DEFR	0FE64h
CC17	DEFR	0FE62h

99/05/08

1			
	CC16	DEFR	0FE60h
	TIREL	DEFR	0FE56h
			0FE54h
	TOREL	DEFR	
	Tl	DEFR	0FE52h
	то	DEFR	0FE50h
	CAPREL	DEFR	0FE4Ah
	т6	DEFR	0FE48h
	T5	DEFR	0FE46h
1		DEFR	0FE44h
	T4		0FE42h
	т3	DEFR	
	т2	DEFR	0FE40h
	PW3	DEFR	0FE36h
	PW2	DEFR	0FE34h
	PW1	DEFR	0FE32h
	PWO	DEFR	0FE30h
	FWO	DELIC	
	; Extended sfi	r area	
	; Excended SI		
	ODP8	DEFR	0F1D6h
	ODP7	DEFR	0F1D2h
	ODP6	DEFR	0F1CEh
	ODP3	DEFR	0F1C6h
		DEFR	0F1C4h
	PICON		0F1C2h
	ODP2	DEFR	
	EXICON	DEFR	0F1C0h
	SOTBIC	DEFR	0F19Ch
	XP3IC	DEFR	0F19Eh
	XP2IC	DEFR	0F196h
	XP1IC	DEFR	0F18Eh
	XPOIC	DEFR	0F186h
			0F17Eh
	PWMIC	DEFR	
	T8IC	DEFR	0F17Ch
	T7IC	DEFR	0F17Ah
	CC31IC	DEFR	0F194h
	CC30IC	DEFR	0F18Ch
	CC29IC	DEFR	0F184h
		DEFR	0F178h
	CC28IC		
	CC27IC	DEFR	0F176h
	CC26IC	DEFR	0F174h
	CC25IC	DEFR	0F172h
	CC24IC	DEFR	0F170h
	CC23IC	DEFR	0F16Eh
	CC22IC	DEFR	0F16Ch
	CC21IC	DEFR	0F16Ah
	CC20IC	DEFR	0F168h
	CC19IC	DEFR	0F166h
	CC18IC	DEFR	0F164h
	CC17IC	DEFR	0F162h
	CC16IC	DEFR	0F160h
	RPOH	DEFR	0F108h
		DEFR	0F106h
	DP1H		0F104h
	DP1L	DEFR	
	DPOH	DEFR	0F102h
	DPOL	DEFR	0F100h
	SSCBR	DEFR	0F0B4h
	SSCRB	DEFR	0F0B2h
	SSCTB	DEFR	OFOBOh
		DEFR	OFOA0h
	ADDAT2		0F056h
	T8REL	DEFR	
	T7REL	DEFR	0F054h
	Т8	DEFR	0F052h
	т7	DEFR	0F050h
	PP3	DEFR	0F03Eh
	PP2	DEFR	0F03Ch
	PP1	DEFR	0F03Ah
	PPI	DEFI	or ophil

9	9/	0	5/(8
2	3:	U.):]	14

and the second state of the second					
		0F038h	AN13	DEFB	P5.13
PPO	DEFR		AN14	DEFB	P5.14
PT3	DEFR	0F036h	AN15	DEFB	P5.15
PT2	DEFR	0F034h			
PT1	DEFR	0F032h	T6EUD	LIT	'AN10'
		0F030h	T5EUD	LIT	'AN11'
PT0	DEFR	0705011	TGIN	LIT	'AN12'
			T5IN	LIT	'AN13'
; Bit names			T4EUD	LIT	'AN14'
CCOIO	DEFB	P2.0			
	DEFB	P2.1	T2EUD	LIT	'AN15'
CC1IO					
CC2IO	DEFB	P2.2	POUTO	DEFB	P7.0
CC3I0	DEFB	P2.3	POUT1	DEFB	P7.1
CC410	DEFB	P2.4		DEFB	P7.2
CC510	DEFB	P2.5	POUT2		
	DEFB	P2.6	POUT3	DEFB	P7.3
CC6IO			CC28I0	DEFB	P7.4
CC710	DEFB	P2.7	CC2910	DEFB	P7.5
CC810	DEFB	P2.8	CC30I0	DEFB	P7.6
CC910	DEFB	P2.9			P7.7
CC10I0	DEFB	P2.10	CC31I0	DEFB	E1.1
	DEFB	P2.11			
CC11IO			CC16I0	DEFB	P8.0
CC12I0	DEFB	P2.12	CC17I0	DEFB	P8.1
CC13I0	DEFB	P2.13	CC18I0	DEFB	P8.2
CC14I0	DEFB	P2.14			
CC15I0	DEFB	P2.15	CC19I0	DEFB	P8.3
		'CC0I0'	CC2010	DEFB	P8.4
EXOIN	LIT		CC21I0	DEFB	P8.5
EX1IN	LIT	'CC1IO'	CC2210	DEFB	P8.6
EX2IN	LIT	'CC210'		DEFB	P8.7
EX3IN	LIT	·CC310'	CC23I0	DEFB	PO. /
EASIN	DII				
	-	52.0			
TOIN	DEFB	P3.0	TOM	DEFB	T01CON.3
TGOUT	DEFB	P3.1		DEFB	TO1CON.6
CAPIN	DEFB	P3.2	TOR		
	DEFB	P3.3	TIM	DEFB	T01CON.11
T3OUT			T1R	DEFB	T01CON.14
T3EUD	DEFB	P3.4	T7M	DEFB	T78CON.3
T2IN	DEFB	P3.7	T7R	DEFB	T78CON.6
T3IN	DEFB	P3.6			T78CON.11
T4IN	DEFB	P3.5	T8M	DEFB	
		P3.8	T8R	DEFB	T78CON.14
SSDI	DEFB				
SSDO	DEFB	P3.9	ACC0	DEFB	CCM0.3
TXD0	DEFB	P3.10	ACC1	DEFB	CCM0.7
RXD0	DEFB	P3.11			
	DEFB	P3.13	ACC2	DEFB	CCM0.11
SSCLK			ACC3	DEFB	CCM0.15
CLKOUT	DEFB	P3.15			
			ACC4	DEFB	CCM1.3
A16	DEFB	P4.0			CCM1.7
A17	DEFB	P4.1	ACC5	DEFB	
	DEFB	P4.2	ACC6	DEFB	CCM1.11
A18			ACC7	DEFB	CCM1.15
A19	DEFB	P4.3	2009-0312-000		
A20	DEFB	P4.4	ACC8	DEFB	CCM2.3
A21	DEFB	P4.5			CCM2.7
A22	DEFB	P4.6	ACC9	DEFB	
			ACC10	DEFB	CCM2.11
A23	DEFB	P4.7	ACC11	DEFB	CCM2.15
AN0	DEFB	P5.0	20013	DEFB	CCM3.3
AN1	DEFB	P5.1	ACC12		
	DEFB	P5.2	ACC13	DEFB	CCM3.7
AN2			ACC14	DEFB	CCM3.11
AN3	DEFB	P5.3	ACC15	DEFB	CCM3.15
AN4	DEFB	P5.4			
AN5	DEFB	P5.5	20016	DEFB	CCM4.3
ANG	DEFB	P5.6	ACC16		
	DEFB	P5.7	ACC17	DEFB	CCM4.7
AN7			ACC18	DEFB	CCM4.11
AN8	DEFB	P5.8	ACC19	DEFB	CCM4.15
AN9	DEFB	P5.9			ANNOUNCES 1973/314 (1970/27
AN10	DEFB	P5.10		Deer	COME 2
	DEFB	P5.11	ACC20	DEFB	CCM5.3
AN11			ACC21	DEFB	CCM5.7
AN12	DEFB	P5.12			

99/05/08 23:03:14		reg16	7b.def		
ACC22 ACC23	DEFB DEFB	CCM5.11 CCM5.15	SSCRIE SSCRIR SSCEIE	DEFB DEFB DEFB	SSCRIC.6 SSCRIC.7 SSCEIC.6
ACC24	DEFB	CCM6.3	SSCEIR	DEFB	SSCEIC.7
ACC24 ACC25	DEFB	CCM6.7	SSCTE	LIT	'SSCTEN'
ACC26	DEFB	CCM6.11	SSCRE	LIT	'SSCREN'
ACC27	DEFB	CCM6.15	SSCPE SSCBE	LIT LIT	'SSCPEN' 'SSCBEN'
ACC28	DEFB	ССМ7.3			
ACC29	DEFB	CCM7.7	CCOIE	DEFB	CCOIC.6
ACC30	DEFB DEFB	CCM7.11 CCM7.15	CCOIR	DEFB	CCOIC.7
ACC31	DEFD	CCM7.15	CC1IE	DEFB	CC1IC.6
T2R	DEFB	T2CON.6	CC1IR	DEFB	CC1IC.7
T2UD	DEFB	T2CON.7	CC2IE	DEFB	CC2IC.6
T2UDE	DEFB	T2CON.8	CC2IR	DEFB	CC2IC.7 CC3IC.6
			CC3IE CC3IR	DEFB	CC3IC.7
T3R	DEFB	T3CON.6	CC4IE	DEFB	CC4IC.6
T3UD	DEFB	T3CON.7	CC4IR	DEFB	CC4IC.7
TJUDE	DEFB	T3CON.8 T3CON.9	CC5IE	DEFB	CC5IC.6
TJOE	DEFB DEFB	T3CON.10	CC5IR	DEFB	CC5IC.7
TJOTL	DEFB	15004.10	CC6IE	DEFB	CC6IC.6
T4R	DEFB	T4CON.6	CC6IR	DEFB	CC6IC.7
T4UD	DEFB	T4CON.7	CC7IE	DEFB DEFB	CC7IC.6 CC7IC.7
T4UDE	DEFB	T4CON.8	CC7IR	DEFB	CC8IC.6
			CC8IE CC8IR	DEFB	CC8IC.7
T5R	DEFB	T5CON.6	CC9IE	DEFB	CC9IC.6
T5UD	DEFB	T5CON.7	CC9IR	DEFB	CC9IC.7
T5UDE	DEFB	T5CON.8	CC10IE	DEFB	CC10IC.6
T5CLR	DEFB	T5CON.14 T5CON.15	CC10IR	DEFB	CC10IC.7
T5SC	DEFB	15CON.15	CC11IE	DEFB	CC11IC.6
T6R	DEFB	T6CON.6	CC11IR	DEFB	CC11IC.7
TOUD	DEFB	T6CON.7	CC12IE	DEFB	CC12IC.6
TOUDE	DEFB	T6CON.8	CC12IR	DEFB	CC12IC.7 CC13IC.6
TGOE	DEFB	T6CON.9	CC13IE CC13IR	DEFB DEFB	CC13IC.7
TGOTL	DEFB	T6CON.10	CC14IE	DEFB	CC14IC.6
T6SR	DEFB	T6CON.15	CC14IR	DEFB	CC14IC.7
100 Sec. 100 Sec. 100	2222	mate 6	CC15IE	DEFB	CC15IC.6
T2IE	DEFB DEFB	T2IC.6 T2IC.7	CC15IR	DEFB	CC15IC.7
T2IR T3IE	DEFB	T3IC.6	CC16IE	DEFB	CC16IC.6
T3IR	DEFB	TJIC.7	CC16IR	DEFB	CC16IC.7
T4IE	DEFB	T4IC.6	CC17IE	DEFB	CC17IC.6
T4IR	DEFB	T4IC.7	CC17IR	DEFB DEFB	CC17IC.7 CC18IC.6
T5IE	DEFB	T5IC.6	CC18IE CC18IR	DEFB	CC18IC.7
T5IR	DEFB	T5IC.7	CC19IE	DEFB	CC19IC.6
TGIE	DEFB	TGIC.6	CC19IR	DEFB	CC19IC.7
T6IR	DEFB	T6IC.7	CC20IE	DEFB	CC20IC.6
ODID	DEFB	CRIC.6	CC20IR	DEFB	CC20IC.7
CRIE	DEFB	CRIC.7	CC21IE	DEFB	CC21IC.6
CRIR	DEFD	CRIC. /	CC21IR	DEFB	CC21IC.7
SOTIE	DEFB	SOTIC.6	CC22IE	DEFB	CC22IC.6 CC22IC.7
SOTIR	DEFB	SOTIC.7	CC22IR	DEFB DEFB	CC23IC.6
SORIE	DEFB	SORIC.6	CC23IE CC23IR	DEFB	CC231C.7
SORIR	DEFB	SORIC.7	CC24IE	DEFB	CC24IC.6
SOEIE	DEFB	SOEIC.6	CC24IE CC24IR	DEFB	CC24IC.7
SOEIR	DEFB	SOEIC.7	CC25IE	DEFB	CC25IC.6
SOTBIE	DEFB	SOTBIC.6 SOTBIC.7	CC25IR	DEFB	CC25IC.7
SOTBIR	DEFB	BUIDIC. /	CC26IE	DEFB	CC26IC.6
CCOTTE	DEFB	SSCTIC.6	CC26IR	DEFB	CC26IC.7
SSCTIE SSCTIR	DEFB	SSCTIC.7	CC27IE	DEFB	CC27IC.6
BOCITIC					
			1		

99/05/08

99/05/08 23:03:14		
CC27IR	DEFB	CC27IC.7
CC28IE	DEFB	CC28IC.6
CC28IR	DEFB	CC28IC.7
CC29IE	DEFB	CC29IC.6
	DEFB	CC291C.7
CC29IR	DEFB	CC30IC.6
CC30IE		
CC30IR	DEFB	CC30IC.7
CC31IE	DEFB	CC31IC.6
CC31IR	DEFB	CC31IC.7
ADCIE	DEFB	ADCIC.6
ADCIR	DEFB	ADCIC.7
ADEIE	DEFB	ADEIC.6
ADEIR	DEFB	ADEIC.7
TOIE	DEFB	TOIC.6
TOIR	DEFB	TOIC.7
T1IE	DEFB	T1IC.6
T1IR	DEFB	T1IC.7
T7IE	DEFB	T7IC.6
T7IR	DEFB	T7IC.7
TSIE	DEFB	T8IC.6
T8IR	DEFB	T8IC.7
	0000	ADGON 7
ADST	DEFB	ADCON.7
ADBSY	DEFB	ADCON.8
ADWR	DEFB	ADCON.9
ADCIN	DEFB	ADCON.10
ADCRQ	DEFB	ADCON.11
ILLBUS	DEFB	TFR.0
ILLINA	DEFB	TFR.1
ILLOPA	DEFB	TFR.2
PRTFLT	DEFB	TFR.3
UNDOPC	DEFB	TFR.7
STKUF	DEFB	TFR.13
STKOF	DEFB	TFR.14
NMI	DEFB	TFR.15
WDTIN	DEFB	WDTCON.0
WDTR	DEFB	WDTCON.1
	5555	60.60¥ 3
SOSTP	DEFB	SOCON.3
SOREN	DEFB	SOCON.4
SOPEN	DEFB	SOCON.5
SOFEN	DEFB	SOCON.6
SOOEN	DEFB	SOCON.7
SOPE	DEFB	SOCON.8
SOFE	DEFB	SOCON.9
SOOE	DEFB	SOCON.10
SOODD	DEFB	S0CON.12
SOBRS	DEFB	SOCON.13
SOLB	DEFB	SOCON.14
SOR	DEFB	SOCON.15
SSCHB	DEFB	SSCCON.4
SSCPH	DEFB	SSCCON.5
SSCPO	DEFB	SSCCON.6
SSCTEN	DEFB	SSCCON.8
SSCREN	DEFB	SSCCON.9
SSCPEN	DEFB	SSCCON.10
SSCBEN	DEFB	SSCCON.11
	DEFB	SSCCON.12
SSCBSY		
SSCMS	DEFB	SSCCON.14
SSCEN	DEFB	SSCCON.15

reg167b.def

	1004110000	
	A45.010	
S	.100	
	- 1998	
	1027	

PTR0	DEFB	PWMCON0.0
PTR1	DEFB	PWMCON0.1
PTR2	DEFB	PWMCON0.2
PTR3	DEFB	PWMCON0.3
PTIO	DEFB	PWMCON0.4
PTI1	DEFB	PWMCON0.5
PTI2	DEFB	PWMCON0.6
PTI3	DEFB	PWMCON0.7
PIEO	DEFB	PWMCON0.8
PIE1	DEFB	PWMCON0.9
PIE2	DEFB	PWMCON0.10
PIE3	DEFB	PWMCON0.11
PIR0	DEFB	PWMCON0.12
PIR1	DEFB	PWMCON0.13
PIR2	DEFB	PWMCON0.14
PIR3	DEFB	PWMCON0.15
PENO		PWMCON1.0
PEN1	DEFB	PWMCON1.1
PEN2	DEFB	PWMCON1.2
PEN3	DEFB	PWMCON1.3
PMO		PWMCON1.4
PM1	DEFB	PWMCON1.5
PM2	DEFB	PWMCON1.6
PM3		PWMCON1.7
PB01		PWMCON1.12
PS2	DEFB	PWMCON1.14
PS3	DEFB	PWMCON1.15
PWMIE	DEFB	PWMIC.6
PWMIR	DEFB	PWMIC.7
I WHILK	DEID	1 WHIC. /
XP3IE	DEFB	XP3IC.6
XP3IR	DEFB	XP3IC.7
XP2IE	DEFB	XP2IC.6
XP2IR	DEFB	XP2IC.7
XP1IE	DEFB	XP1IC.6
XP1IR	DEFB	XP1IC.7
XPOIE	DEFB	XPOIC.6
XPOIR	DEFB	XPOIC.7

B.5 42V Bus CAN Node 1

On the next page starts the code for the 42V bus CAN node 1. The files for the node are as follows.

- 1. comp142.bat
- 2. main142.asm
- 3. cnmod142.asm
- 4. canmo142.asm
- 5. cnint142.asm
- 6. atod142.asm
- 7. tmrs142.asm
- 8. linker.lnv
- 9. Reg167b.def



al66 main142.asm al66 cnmod142.asm al66 canmol42.asm al66 canint142.asm al66 atod142.asm al66 tmrs142.asm ll66 LINK main142.obj cnmod142.obj canmol42.obj cnint142.obj atod142.obj tmrs142.obj TO locatein.lno ll66 @linker.lnv ihex166 -i16 locate.out -o main142.hex



main142.asm



;; Initialize CAN Bus SEGMENTED ; Call the CAN initialization function CALL canin SEXTEND ;; End of CAN Bus Initialization SEXTSFR ; CAN USE ALL internal RAM for Stack \$EXTSSK meto: SEXTMEM ; just loop here waiting NOP \$NOMOD166 NOP \$STDNAMES(reg167b.def) JMP meto SSYMBOLS RET ; return main ENDP NAME main ; define a common register area of 16 register mainseg ENDS RBANK1 COMREG R0-R15 ; codesegment that contains reset int pointer startupsec SECTION CODE : default stack size of 256 Words SSKDEF 4 ; reset interrupt number is zero at 0h sysreset PROC TASK INTNO=0H ; forces next instruction to be located at Oh ORG 000H ASSUME DPP3:SYSTEM ; installs a pointer to the startup routine JMP start ; return from interrupt RETT EXTERN canin: FAR ; Can function sysreset ENDP ; external atod initialization EXTERN atod initialize:FAR startupsec ENDS EXTERN atod_timer_initialize:FAR END mainseg SECTION CODE main PROC FAR ; disable the watchdog timer start: DISWDT ; Globally Enable Interrupts both global BSET IEN ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh ; end initialization EINIT ;; End of external memory bus initialization ;; Initialize the Data Page pointers for this section ; make DPP3 point to system MOV DPP3, #03h ;; End of Data Page Pointer Initialization ;; Make the direction of Port 2 to output MOV DP2, ONES ;; Make sure Port 2 is in push/pull mode MOV ODP2, ONES ;; Initialize The Stack ;; The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #0FBFFh the highest ;; byte that the stack pointers can point to is #OFBFEh MOV STKUN, #0FBFEh; Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP, #0FBFEh ; Set the Stack Pointer :: End of Stack Initialization ;; Initialize the Analog to Digital Converter CALL atod_initialize; atod ;; End of A/D initialization ;; Initialize A/D timer CALL atod_timer_initialize; timers ;; End of A/D timer initialization



cnmod142.asm



\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM	RET canin ENDP setall PROC FAR ; This Procedure sets all of the Mess objs invalid ;; by using a counter it counts up to 15 and initializes all of the message
\$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS	;; objects along the way. PUSH R2 PUSH R4
NAME canmod	PUSH R5 AND R5, ZEROS
RBANK1COMREG R0-R15; define a common register area of 16 registersGLOBALcanin; The function must be declared Global at the; beginning of the module	AND R2,ZEROS OR R2,#0EF10h ; Set pointer to MO1 AND R4,ZEROS
EXTERN canmocfg:FAR ; configures specific Message objects	OR R4, #5555h ; Set R4 to make MObs invalid
ASSUME DPP3:SYSTEM	nextreg:MOV [R2],R4 ; make all message objects invalid ADD R2,#10h
canfunc SECTION CODE ; codesegment that contains reset int pointer	CMPI1 R5,#0Fh JMPA CC_NZ,nextreg ;
canin PROC FAR PUSH R0 PUSH R1	POP R5 POP R4 POP R2 RET
;; set all of the CAN control registers AND C1CSR,ZEROS ; set control register to zero MOV R1, #0043h ; Set IE and INIT bits OR C1CSR,R1 ; set control register to R1's value	setall ENDP canfunc ENDS END
AND C1BTR, ZEROS ; set Bit timing register to zero MOV R1, #03447h ; set for 125k operation OR C1BTR, R1 ; set Bit timing register parameters	
AND C1GMS, ZEROS ; set Global Mask short register to zero MOV R1, #0FFFFh ; EOFF is what DAVE initialize OR C1GMS, R1 ; set GMS	
AND ClUGML, ZEROS ; set Upper global mask long to zero MOV R1, #0FFFFh OR ClUGML, R1	
MOV R1, #0F8FFh AND C1LGML, ZEROS OR C1LGML, R1 ; lower global mask	
AND ClUMLM, ZEROS OR ClUMLM, R1 ; upper mask of last register AND ClLMLM, ZEROS	
OR C1LMLM, R1 ; lower mask of last register	
CALL setall ; sets all of the CAN registers to off	
CALL canmocfg ; Configures specific Message Objects	
;; Setup CAN interrupt and Initialize CAN module EXTR #4 AND XPOIC, ZEROS ; configure CAN interrupt control Register	
AND R0,ZEROS OR R0,#0073h ; enable interrupt, level is 10 group is 2 OR XPOIC,R0 ; Configure CAN interrupt Control Register	
AND R1, ZEROS OR R1, #00041h ; crashes if I clear the CPU access to the BTR XOR C1CSR, R1 ; end initialize CAN interrupt POP R1 POP R0	

99/05/09 12:22:54		canmo14	2.asm	e state production d'anne No de la constante de la constante No de la constante de la consta	
\$SEGMENTED					's Control register
\$EXTEND				ADD R2, #2h ;	point to Upper Arbitration register
				AND R3, ZEROS	; set R6 to zero
\$EXTSFR				OR R3, #06003h ;	The number is the Message ID for Message Object 3
\$EXTMEM				MOV [R2], R3 ;	message id = 0
\$NOMOD166					Point to the Lower Arbitration Register
\$STDNAMES(reg167b.def)					standard Message object so lowerarb = 0h
\$SYMBOLS				AND R1, ZEROS	
					; put 000h into first data byte and set to receive
NAME canmo					; Databyte(0) = 0 and Set to receive and 3 bytes 0
RBANK1 COMREC	G R0-R15	; declare bank of 16 global registers		MOV MCD_M3,R1	; $Databyte(0) = 0$ and set to receive and 5 bytes 0
GLOBAL canmoo		A AID'	f data		
GIODAL CUILING				MOV DATA_M3, ZEROS ;	; Fill the Data of the MO with Zeros
can_module	SECTION CODE			;; Initialize Message Obj	
can_module	SECTION CODE			MOV R2, #MCR_M4 ;	; start of Message Object 4
				AND R1, ZEROS	
ASSUME DPP3:	SYSTEM			OR R1, #5595h	
					4's Control register
canmocfg PR	OC FAR				; point to Upper Arbitration register
PUSH	R1				; set R6 to zero
PUSH	R2			AND R3, ZEROS	; Set Ro to zero ; The number is the Message ID for Message Object 4
PUSH	R3				
:: N	ow set specific CAN	control Registers		not (mo)/mo	; message id = 0
	nitialize message of	piect 1			; Point to the Lower Arbitration Register
,, , ,	nitializing this obj	ect to be invalid does or removing the code until		MOV [R2], ZEROS	; standard Message object so lowerarb = Oh
;; the comment "Setup CAN interrupt and Initialize" does				AND R1, ZEROS	
;; L	ne conument secup cr	e occurrance of the interrupt for the CAN system		OR R1, #0038h	; put OAAh into first data byte and set to receive
		; start of Message Object 1		MOV MCD_M4,R1	; Databyte(0) = 0 and Set to receive and 3 bytes o
		; start of Message object i	f data		
	, ZEROS	- I	r ducu	MOV DATA_M4, ZEROS	; fill the data of the MO with ZEROS
OR R	1, #5599h	; Generate a Receive Interrupt if this message object ac		NOV DATA_NA, DEROD	
tivates MOV [R2],R1 ; set MO1's Control register					
	(112)/112 / 222				
מתא	R2,#2h	; point to Upper Arbitration register		;; Initialize Message Ob;	
	, ZEROS	; set R3 to		MOV R2, #MCR_M5	; start of Message Object 5
		message id for message object 1		AND R1, ZEROS	
		; message id = #0003h		OR R1, #5595h	i
	[R2],R3	; Point to the Lower Arbitration Register			4's Control register
	R2, #2h				; point to Upper Arbitration register
MOV	[R2], ZEROS	; standard Message object so lowerarb = Oh		AND R3, ZEROS	; set R6 to zero
AND	R1, ZEROS				; The number is the Message ID for Message Object 5
OR R	1, #0030h	; put OAAh into first data byte and set to receive			
	D_M1,R1	; Databyte(0) = 0 and Set to receive and 3 bytes of data			; message id = 0
	DATA_M1, ZEROS	; fill the Data of the MO with Zeros		ADD R2, #2h	; Point to the Lower Arbitration Register
110 0	DATA_AT, BERED	 E E E O DESE SALARAS PARTI STATE MART ANALYSIS STATEMENTS 		MOV [R2], ZEROS	; standard Message object so lowerarb = Oh
т.	nitialize Message O	piect 2		AND R1, ZEROS	
		; start of Message Object 2		OR R1, #0038h	; put OAAh into first data byte and set to receive
	R2, #MCR_M2	; Start of Message object 2		MOV MCD_M5,R1	; Databyte(0) = 0 and Set to receive and 3 bytes o
	R1, ZEROS	DESTRICT INTERPOLICE anabled	f data		 Press and a second s Second second se
OR F	1, #5599h	; RECEIVE INTERRUPT enabled	I uata	MOV DATA_M5, ZEROS	; fill the data of the MO with ZEROS
MOV	[R2],R1 ; set M	02's Control register		MOV DATA_MJ, ZEROS	, IIII the data of the howith ballo
ADD	R2,#2h	; point to Upper Arbitration register			
	R3, ZEROS	; set R6 to zero			Anna An
	R3, #04003h	; The number is the Message ID for Message Object 2		;; Initialize Message Ob	
		; message id = 0		MOV R2, #MCR_M6	; start of Message Object 6
	[R2],R3 R2, #2h	; Point to the Lower Arbitration Register		AND R1, ZEROS	
		; standard Message object so lowerarb = 0h		OR R1, #5595h	;
	[R2], ZEROS	, beanaard hoobage enjett it senether			4's Control register
	R1, ZEROS	and inte first data buts and got to receive			; point to Upper Arbitration register
	R1, #0030h	; put 000h into first data byte and set to receive		AND R3, ZEROS	; set R6 to zero
MOV	MCD_M2,R1	; Databyte(0) = 0 and Set to receive and 3 bytes of da			; The number is the Message ID for Message Object 6
ta					; message id = 0
MOV	DATA_M2, ZEROS	; Fill the Data of the MO with Zeros			; Point to the Lower Arbitration Register
	Initialize Message O	bject 3			; standard Message object so lowerarb = 0h
	R2, #MCR_M3	; start of Message Object 3		AND R1, ZEROS	
	R1, ZEROS			OR R1, #0038h	; put OAAh into first data byte and set to receive
	R1, #5595h	; Generate a receive interrupt if this message object ac		MOV MCD_M6,R1	; Databyte(0) = 0 and Set to receive and 3 bytes o
	x1, ποογοιι	· · · · · · · · · · · · · · · · · · ·	f data		
tivates					



canmo142.asm



MOV DATA_M6, ZEROS

; fill the data of the MO with ZEROS

POP R3 POP R2 POP R1 RET canmocfg ENDP

can_module ENDS END



\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS

NAME canint RBANK1 COMREG R0-R15

; declare bank of 16 global registers

ASSUME DPP3:SYSTEM

can_interrupts SECTION CODE

can_receive_interrupt PROC TASK INTNO=040h ORG 0100h CALL can_receive_interrupt_handler RETI can_receive_interrupt ENDP

can_receive_interrupt_handler PROC FAR PUSH R0

PUSH R1 PUSH R2 ; Read the CAN interrupt ID buffer MOVB RLO, INTID ; See if the interrupt came from M01 CMPB RLO, #03h JMP cc_Z, message_one_interrupt; if interrupt from M01 handle MOV R1, #05555h MOV R2, #05599h MOV MCR_M2, R1 MOV R0, DATA_M2 MOV MCR_M2, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch MOV R2, MCR_M6 MOV MCR_M6, R1 MOV DATA_M6, R0 MOV MCR_M6, R2 CMP R0, #01h JMP cc_NZ, turn_off_heated_rear_window BSET P2.1 JMP exit_function

turn_off_heated_rear_window: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.1

JMP exit_function

> MOV R2, MCR_M5 MOV MCR_M5, R1

cnint142.asm

MOV DATA_M5, R0

MOV MCR_M5, R2 CMP R0, #01h JMP cc_NZ, turn_heater_off BSET P2.0 JMP exit_function

turn_heater_off: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.0 exit_function: MOV R2, #0EFFFh AND C1CSR, R2 POP R2 POP R1 POP R0 RET can_receive_interrupt_handler ENDP can_interrupts ENDS END





atod142.asm



PUSH R3 SEGMENTED PUSH R4 SEXTEND PUSH MDH SEXTSFR PUSH MDL : CAN USE ALL internal RAM for Stack SEXTSSK MOV R2, ADDAT ; This is so we can isolate the A/D channel from whi SEXTMEM MOV RO, R2 \$NOMOD166 ch the data is coming \$STDNAMES(reg167b.def) ; This is so we can isolate the A/D data MOV R3, R2 SSYMBOLS AND R3, #03FFh ; This isolates the A/D data No Scaling to be done on Microcontroller MOV R4, #01h ; AND RO, #0F000h ; The channel information is located in the upper nibble name atod CMP R0, #01000h ; See if the information is coming from Channel 1 of the A/ D ASSUME DPP3:SYSTEM JMP cc_Z, Rear_Seat_Heater_current RBANK1 COMREG R0-R15 GLOBAL atod_initialize : This bit pattern deactives MCRs MOV R0, #05555h MOV R1, MCR_M3 ; SAVE the Configuration of the MCR ;; This A/D is set up to measure the current in two different ; Kill the Message Control Register MOV MCR_M3, R0 ;; loads. Because this software is to be used as part of ;; 42volt bus node 1, it uses the names of the loads that ; This multiplication returns the actual value of the current flowing throu ;; that node is supposed to control. gh the transistor ;; The analog to digital converter uses Port 5 MUL R3, R4 NOP MOV DATA_M3, MDL ; for real atod_setup SECTION CODE MOV MCR M3, R1 BSET T3R atod_initialize PROC FAR JMP exit_routine ;; Initialize variables ;; This below line of code setups up the A/D converter Rear_Seat_Heater_current: ;; for 2 channels and single conversion. ;; It is also set for "Wait for read mode" ; This bit pattern deactives MCRs MOV R0, #05555h ;; so the converter will wait for the user program to read ; SAVE the Configuration of the MCR MOV R1, MCR_M4 ;; the buffer before processing the next channel. ; Kill the Message Control Register MOV MCR_M4, R0 : setup A/D control register MOV ADCON, #0A221h ;; This test code counts out on Port 2 and if it doesn't ;; Then that means that the A/D and timer aren't working MOV R0, #04h ;test code ;; Set the channel to which the data should be written :test code ADD P2, R0 ;; when the first "A/D is done" interrupt occurs MUL R3, R4 ;; The below code sets up the A/D's Interrupt control register ;; The A/D is setup to have a group of 2 and a level of 10 NOP MOV DATA_M4, MDL ; for testing purposes MOV ADCIC, #006Fh MOV MCR M4, R1 RET atod initialize ENDP exit_routine: atod_setup ENDS POP MDL POP MDH atod_handlers SECTION CODE POP R4 atod_handler PROC TASK INTNO=028h POP R3 ORG 0A0H POP R2 CALL atod_function POP R1 RETT POP RO atod handler ENDP RET atod_function ENDP atod function PROC FAR atod_handlers ENDS ;; this function works by seeing if the converter is converting ;; for the heater_measurement. If the bit is set, then END ;; the bit gets cleared and the IP jumps to where the ;; value in the converter is moved into the heater_current ;; variable. ;; otherwise the bit gets set and the value is moved into ;; the heated_rear_window_current variable PUSH RO PUSH R1 PUSH R2



tmrs142.asm

; These are assembler controls **\$SEGMENTED** \$EXTEND \$EXTSFR \$EXTMEM \$EXTINSTR \$NOMOD166 \$STDNAMES(reg167b.def) ; Assembler controls end here \$SYMBOLS NAME timer_functions ASSUME DPP3:SYSTEM RBANK1 COMREG R0-R15 GLOBAL atod_timer_initialize atod_timer SECTION CODE atod_timer_initialize PROC FAR ; setup Core Timer T3 MOV T3CON, #0004h MOV T3IC, #002Bh ; Make the value in the counter equal to zero MOV T3, #0000h ; enable the timer interrupt BSET T3IE BSET T3R ; start the timer RET atod_timer_initialize ENDP atod_interrupt PROC TASK INTNO=023h ORG 08Ch CALL atod_timer_handler RETI atod_interrupt ENDP atod_timer_handler PROC FAR ; stop the timer BCLR T3R ; start an A/D conversion BSET ADST RET atod_timer_handler ENDP atod_timer ENDS END



LOCATE locatein.lno (GENERAL) IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (040000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO locate.out linker.lnv



99/05/08 23:03:14

CONTRACTOR OF A					
		************	P8	DEFR	0FFD4h
;*********				DEFR	0FFD2h
	6/b.der	1.10 12/18/97	DP7 P7	DEFR	0FFD0h
;**			DP6	DEFR	OFFCEh
;** Register	definitio	ons for the SAB C167	P6	DEFR	0FFCCh
;** This file	e contains	s all SFR names and BIT names	DP4	DEFR	OFFCAh
;** This file	e can be s	supplied to rm166 and a166 (STDNAMES control)	P4	DEFR	0FFC8h
; * * * * * * * * * * *	*******	**************************************		DEFR	0FFC6h
TRUE	DEFB	OFF20h.0, RW	DP3		
NODE142	DEFB	OFF2Oh.1, RW	P3	DEFR	0FFC4h
			DP2	DEFR	0FFC2h
C1CSR	DEFA	OEFOOh	P2	DEFR	OFFCOh
INTID	DEFA	0EF02h	SSCCON	DEFR	0FFB2h
C1BTR	DEFA	0EF04h	SOCON	DEFR	OFFBOh
CIGMS	DEFA	0EF06h	WDTCON	DEFR	OFFAEh
CIUGML	DEFA	0EF08h	TFR	DEFR	OFFACh
CILGML	DEFA	OEFOAh	P5	DEFR	0FFA2h
CIUMLM	DEFA	OEFOCh	ADCON	DEFR	OFFA0h
	DEFA	OEFOEh	TIIC	DEFR	0FF9Eh
C1LMLM	DEFA	0EF10h	TOIC	DEFR	0FF9Ch
MCR_M1		0EF20h	ADEIC	DEFR	0FF9Ah
MCR_M2			ADCIC	DEFR	0FF98h
MCR_M3	DEFA	0EF30h	CC15IC	DEFR	0FF96h
MCR_M4	DEFA	0EF40h	CC14IC	DEFR	0FF94h
MCR_M5	DEFA	0EF50h	CC13IC	DEFR	0FF92h
MCR_M6		0EF60h	CC12IC	DEFR	0FF90h
MCR_M7	DEFA	0EF70h	CC11IC	DEFR	0FF8Eh
MCR_M8	DEFA	0EF80h			0FF8Ch
MCR_M9	DEFA	0EF90h	CC10IC	DEFR	
MCR_MA	DEFA	OEFAOh	CC9IC	DEFR	0FF8Ah
MCR_MB	DEFA	0EFB0h	CC8IC	DEFR	0FF88h
MCR_MC	DEFA	0EFC0h	CC7IC	DEFR	0FF86h
MCR_MD	DEFA	0EFD0h	CC6IC	DEFR	0FF84h
MCR_ME	DEFA	0EFE0h	CC5IC	DEFR	0FF82h
MCR_MF	DEFA	OEFFOh	CC4IC	DEFR	0FF80h
	DEFA	0EF16h	CC3IC	DEFR	0FF7Eh
MCD_M1	DEFA	0EF26h	CC2IC	DEFR	0FF7Ch
MCD_M2	DEFA	0EF36h	CC1IC	DEFR	0FF7Ah
MCD_M3			CCOIC	DEFR	0FF78h
MCD_M4	DEFA	0EF46h	SSCEIC	DEFR	0FF76h
MCD_M5	DEFA	0EF56h	SSCRIC	DEFR	0FF74h
MCD_M6	DEFA	0EF66h	SSCTIC	DEFR	0FF72h
MCD_M7	DEFA	0EF76h	SOEIC	DEFR	0FF70h
MCD_M8	DEFA	0EF86h	SORIC	DEFR	0FF6Eh
MCD_M9	DEFA	0EF96h	SOTIC	DEFR	0FF6Ch
MCD_MA	DEFA	0EFA6h		DEFR	0FF6Ah
MCD_MB	DEFA	0EFB6h	CRIC		
MCD_MC	DEFA	0EFC6h	TGIC	DEFR	0FF68h
MCD_MD	DEFA	0EFD6h	T5IC	DEFR	0FF66h
MCD_ME	DEFA	0EFE6h	T4IC	DEFR	0FF64h
DATA_M1	DEFA	0EF18h	T3IC	DEFR	0FF62h
DATA M2	DEFA	0EF28h	T2IC	DEFR	0FF60h
DATA_M3	DEFA	0EF38h	CCM3	DEFR	0FF58h
DATA_M4	DEFA	0EF48h	CCM2	DEFR	0FF56h
DATA_M5	DEFA	0EF58h	CCM1	DEFR	0FF54h
DATA_M6	DEFA	0EF68h	CCM0	DEFR	0FF52h
· 그 같아. 것 않는 것 등 것 등 중 것 있 것	DEFA	0EF78h	TOICON	DEFR	0FF50h
DATA_M7		0EF88h	T6CON	DEFR	0FF48h
DATA_M8	DEFA		TSCON	DEFR	0FF46h
DATA_M9	DEFA	0EF98h	T4CON	DEFR	0FF44h
DATA_MA	DEFA	0EFA8h	TJCON	DEFR	0FF42h
DATA_MB	DEFA	0EFB8h	T2CON	DEFR	0FF40h
DATA_MC	DEFA	0EFC8h	PWMCON1	DEFR	0FF32h
DATA_MD	DEFA	0EFD8h	PWMCON1	DEFR	0FF30h
DATA_ME	DEFA	0EFE8h		DEFR	0FF28h
			CCM7		0FF26h
			CCM6	DEFR	
			CCM5	DEFR	0FF24h
DP8	DEFR	0FFD6h	CCM4	DEFR	0FF22h

99/05/08 23:03:14			
T78CON	DEFR	0FF20h	
P1H	DEFR	0FF06h	
P1L	DEFR	0FF04h	
POH	DEFR	0FF02h	
POL	DEFR	OFFOOh	
PECC7	DEFR	OFECEh	
PECC6	DEFR	OFECCh	
PECC5	DEFR	OFECAh	
PECC4	DEFR	0FEC8h	
PECC3	DEFR	0FEC6h	
PECC2	DEFR	0FEC4h 0FEC2h	
PECC1	DEFR	0FEC0h	
PECC0 SRCP0	DEFA	OFCEOh	
DSTP0	DEFA	0FCE2h	
SRCP1	DEFA	0FCE4h	
DSTP1	DEFA	0FCE6h	
SRCP2	DEFA	0FCE8h	
DSTP2	DEFA	OFCEAh	
SRCP3	DEFA	OFCECh	
DSTP3	DEFA	OFCEEh	
SRCP4	DEFA	OFCFOh	
DSTP4	DEFA	0FCF2h	
SRCP5	DEFA	0FCF4h	
DSTP5	DEFA	0FCF6h 0FCF8h	
SRCP6	DEFA	OFCFAh	
DSTP6	DEFA	0FCFCh	
SRCP7 DSTP7	DEFA	OFCFEh	
SOBG	DEFR	0FEB4h	
SORBUF	DEFR	OFEB2h,	r
SOTBUF	DEFR		w
WDT	DEFR	OFEAEh,	r
ADDAT	DEFR	0FEA0h	
CC15	DEFR	0FE9Eh	
CC14	DEFR	0FE9Ch	
CC13	DEFR	0FE9Ah	
CC12	DEFR	0FE98h	
CC11	DEFR	0FE96h	
CC10	DEFR	0FE94h	
CC9	DEFR	0FE92h	
CC8	DEFR	0FE90h 0FE8Eh	
CC7	DEFR	0FE8Ch	
CC6 CC5	DEFR	0FE8Ah	
CC4	DEFR	0FE88h	
CC3	DEFR	0FE86h	
CC2	DEFR	0FE84h	
CC1	DEFR	0FE82h	
CCO	DEFR	0FE80h	
CC31	DEFR	0FE7Eh	
CC30	DEFR	0FE7Ch	
CC29	DEFR	0FE7Ah	
CC28	DEFR	0FE78h	
CC27	DEFR	0FE76h	
CC26	DEFR	0FE74h	
CC25	DEFR	0FE72h	
CC24	DEFR	0FE70h	
CC23	DEFR	0FE6Eh	
CC22	DEFR	0FE6Ch 0FE6Ah	
CC21	DEFR	0FE68h	
CC20	DEFR DEFR	0FE66h	
CC19 CC18	DEFR	0FE64h	
CC17	DEFR	0FE62h	
	DEFIX		

reg167	b.def		
	CC16 T1REL T0REL T1 T0 CAPREL T6 T5 T4 T3 T2	DEFR DEFR DEFR DEFR DEFR DEFR DEFR DEFR	0FE60h 0FE56h 0FE54h 0FE52h 0FE50h 0FE48h 0FE48h 0FE46h 0FE42h 0FE42h
	PW3 PW2 PW1 PW0 : Extended	DEFR DEFR DEFR DEFR	OFE36h OFE34h OFE32h OFE30h
	ODP8 ODP7 ODP6 ODP3 PICON ODP2 EXICON	DEFR DEFR DEFR DEFR DEFR DEFR DEFR	0F1D6h 0F1D2h 0F1CEh 0F1C6h 0F1C4h 0F1C2h 0F1C0h
	SOTBIC XP3IC XP2IC XP1IC XP0IC PWMIC T8IC T7IC	DEFR DEFR DEFR DEFR DEFR DEFR DEFR	0F19Ch 0F19Eh 0F196h 0F18Eh 0F186h 0F17Eh 0F17Ch 0F17Ah
	CC311C CC301C CC291C CC281C CC271C CC271C CC261C CC251C	DEFR DEFR DEFR DEFR DEFR DEFR	0F194h 0F18Ch 0F184h 0F178h 0F176h 0F174h 0F172h
	CC24IC CC23IC CC22IC CC21IC CC20IC CC19IC CC19IC CC18IC CC17IC	DEFR DEFR DEFR DEFR DEFR DEFR DEFR	0F170h 0F16Eh 0F16Ch 0F16Ah 0F168h 0F166h 0F164h 0F162h
	CC16IC RPOH DP1H DP0L DP0H DP0L SSCBR	DEFR DEFR DEFR DEFR DEFR DEFR	0F160h 0F108h 0F106h 0F104h 0F102h 0F100h 0F0B4h
	SSCRB SSCTB ADDAT2 T&REL T7REL T8 T7 PP3 PP2	DEFR DEFR DEFR DEFR DEFR DEFR DEFR DEFR	0F0B2h 0F0B0h 0F0A0h 0F056h 0F052h 0F052h 0F050h 0F03Eh 0F03Ch
	PP1	DEFR	0F03Ah

23:03:14		
PP0 PT3	DEFR DEFR	0F038h 0F036h
PT2 PT1	DEFR DEFR	0F034h 0F032h
PTO	DEFR	0F030h
; Bit names CC0IO	DEFB	P2.0
CC110	DEFB	P2.1
CC210 CC310	DEFB	P2.2 P2.3
CC410	DEFB	P2.4
CC510	DEFB	P2.5
CC610 CC710	DEFB DEFB	P2.6 P2.7
CC810	DEFB	P2.8
CC910	DEFB	P2.9
CC10IO CC11IO	DEFB DEFB	P2.10 P2.11
CC1210	DEFB	P2.12
CC13I0	DEFB	P2.13
CC14I0 CC15I0	DEFB DEFB	P2.14 P2.15
EXOIN	LIT	, CC010,
EX1IN	LIT	'CC1I0' 'CC2I0'
EX2IN EX3IN	LIT LIT	'CC310'
TOIN	DEFB	P3.0 P3.1
T6OUT CAPIN	DEFB DEFB	P3.1 P3.2
T3OUT	DEFB	P3.3
T3EUD T2IN	DEFB DEFB	P3.4 P3.7
TJIN	DEFB	P3.6
T4IN	DEFB	P3.5
SSDI SSDO	DEFB DEFB	P3.8 P3.9
TXD0	DEFB	P3.10
RXD0	DEFB	P3.11 P3.13
SSCLK CLKOUT	DEFB DEFB	P3.15 P3.15
A16	DEFB	P4.0 P4.1
A17 A18	DEFB	P4.1 P4.2
A19	DEFB	P4.3
A20 A21	DEFB	P4.4 P4.5
A21 A22	DEFB	P4.6
A23	DEFB	P4.7
AN0	DEFB	P5.0
AN1 AN2	DEFB	P5.1 P5.2
AN3	DEFB	P5.3
AN4	DEFB	P5.4
AN5 AN6	DEFB DEFB	P5.5 P5.6
AN7	DEFB	P5.7
AN8	DEFB	P5.8 P5.9
AN9 AN10	DEFB DEFB	P5.9 P5.10
AN11	DEFB	P5.11
AN12	DEFB	P5.12

99/05/08

reg1	67b	.def
1051	010	·uci

.def		
AN13 AN14 AN15 T6EUD T5EUD T6IN T5IN T4EUD T2EUD	DEFB DEFB LIT LIT LIT LIT LIT LIT	P5.13 P5.14 P5.15 'AN10' 'AN11' 'AN12' 'AN13' 'AN13' 'AN14' 'AN15'
POUTO POUT1 POUT2 POUT3 CC28IO CC29IO CC30IO CC31IO	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P7.0 P7.1 P7.2 P7.3 P7.4 P7.5 P7.6 P7.6 P7.7
CC16IO CC17IO CC18IO CC19IO CC20IO CC21IO CC22IO CC23IO	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P8.0 P8.1 P8.2 P8.3 P8.4 P8.5 P8.6 P8.7
TOM TOR T1M T1R T7M T7R T8M T8R	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T01CON.3 T01CON.6 T01CON.11 T01CON.14 T78CON.3 T78CON.6 T78CON.11 T78CON.14
ACCO	DEFB	CCM0.3
ACC1	DEFB	CCM0.7
ACC2	DEFB	CCM0.11
ACC3	DEFB	CCM0.15
ACC4	DEFB	CCM1.3
ACC5	DEFB	CCM1.7
ACC6	DEFB	CCM1.11
ACC7	DEFB	CCM1.15
ACC8	DEFB	CCM2.3
ACC9	DEFB	CCM2.7
ACC10	DEFB	CCM2.11
ACC11	DEFB	CCM2.15
ACC12	DEFB	CCM3.3
ACC13	DEFB	CCM3.7
ACC14	DEFB	CCM3.11
ACC15	DEFB	CCM3.15
ACC16	DEFB	CCM4.3
ACC17	DEFB	CCM4.7
ACC18	DEFB	CCM4.11
ACC19	DEFB	CCM4.15
ACC20	DEFB	CCM5.3
ACC21	DEFB	CCM5.7

3

99/05/08 23:03:14			rag167	h def		
23.03.14			reg167	U.uel		
	tottototot			SSCRIE	DEFB	SSCRIC.6
ACC22	DEFB	CCM5.11		SSCRIR	DEFB	SSCRIC.7
ACC23	DEFB	CCM5.15		SSCEIE	DEFB	SSCEIC.6
	DDDD	CCM6.3		SSCEIR	DEFB	SSCEIC.7
ACC24	DEFB DEFB	CCM6.7		SSCTE	LIT	'SSCTEN'
ACC25	DEFB	CCM6.11		SSCRE	LIT	'SSCREN'
ACC26 ACC27	DEFB	CCM6.15		SSCPE	LIT	'SSCPEN'
ACC27	DELD	CONV.15		SSCBE	LIT	'SSCBEN'
ACC28	DEFB	CCM7.3				
ACC29	DEFB	CCM7.7				
ACC30	DEFB	CCM7.11		CCOIE	DEFB	CCOIC.6
ACC31	DEFB	CCM7.15		CCOIR	DEFB	CCOIC.7
				CC1IE	DEFB	CC1IC.6
T2R	DEFB	T2CON.6		CC1IR	DEFB	CC1IC.7
T2UD	DEFB	T2CON.7		CC2IE	DEFB	CC2IC.6
T2UDE	DEFB	T2CON.8		CC2IR	DEFB	CC2IC.7
An el al anticipation el la construcción de				CC3IE	DEFB	CC3IC.6
T3R	DEFB	T3CON.6		CC3IR	DEFB	CC3IC.7
T3UD	DEFB	T3CON.7		CC4IE	DEFB	CC4IC.6
T3UDE	DEFB	T3CON.8		CC4IR	DEFB DEFB	CC4IC.7 CC5IC.6
T3OE	DEFB	T3CON.9		CC5IE	DEFB	CC5IC.7
T3OTL	DEFB	T3CON.10		CC5IR	DEFB	CC6IC.6
				CC6IE	DEFB	CC6IC.7
T4R	DEFB	T4CON.6		CC6IR	DEFB	CC7IC.6
T4UD	DEFB	T4CON.7		CC7IE CC7IR	DEFB	CC7IC.7
T4UDE	DEFB	T4CON.8		CC8IE	DEFB	CC8IC.6
				CC8IR	DEFB	CC8IC.7
T5R	DEFB	T5CON.6		CC9IE	DEFB	CC9IC.6
T5UD	DEFB	T5CON.7		CC9IR	DEFB	CC9IC.7
T5UDE	DEFB	T5CON.8		CC10IE	DEFB	CC10IC.6
T5CLR	DEFB	T5CON.14		CC10IR	DEFB	CC10IC.7
T5SC	DEFB	T5CON.15		CC11IE	DEFB	CC11IC.6
	0000	MCOON 6		CC11IR	DEFB	CC11IC.7
T6R	DEFB	T6CON.6 T6CON.7		CC12IE	DEFB	CC12IC.6
T6UD	DEFB	T6CON. 8		CC12IR	DEFB	CC12IC.7
T6UDE	DEFB DEFB	T6CON. 9		CC13IE	DEFB	CC13IC.6
T60E T60TL	DEFB	T6CON.10		CC13IR	DEFB	CC13IC.7
T6SR	DEFB	T6CON.15		CC14IE	DEFB	CC14IC.6
TOSK	DEFD	100000.15		CC14IR	DEFB	CC14IC.7
T2IE	DEFB	T2IC.6		CC15IE	DEFB	CC15IC.6
T2IR	DEFB	T2IC.7		CC15IR	DEFB	CC15IC.7
T3IE	DEFB	T3IC.6		CC16IE	DEFB	CC16IC.6
T3IR	DEFB	T3IC.7		CC16IR	DEFB	CC16IC.7
T4IE	DEFB	T4IC.6		CC17IE	DEFB	CC17IC.6
T4IR	DEFB	T4IC.7		CC17IR	DEFB	CC17IC.7
T5IE	DEFB	T5IC.6		CC18IE	DEFB	CC18IC.6
T5IR	DEFB	T5IC.7		CC18IR	DEFB	CC18IC.7
T6IE	DEFB	T6IC.6		CC19IE	DEFB	CC19IC.6
T6IR	DEFB	T6IC.7		CC19IR	DEFB	CC19IC.7
				CC20IE	DEFB	CC20IC.6
CRIE	DEFB	CRIC.6		CC20IR	DEFB	CC20IC.7
CRIR	DEFB	CRIC.7		CC21IE	DEFB	CC21IC.6
24536-5243				CC21IR	DEFB	CC21IC.7
SOTIE	DEFB	SOTIC.6		CC22IE	DEFB	CC22IC.6
SOTIR	DEFB	SOTIC.7		CC22IR	DEFB	CC22IC.7
SORIE	DEFB	SORIC.6		CC23IE	DEFB	CC23IC.6 CC23IC.7
SORIR	DEFB	SORIC.7		CC23IR	DEFB DEFB	CC24IC.6
SOEIE	DEFB	SOEIC.6		CC24IE	DEFB	CC24IC.8
SOEIR	DEFB	SOEIC.7		CC24IR	DEFB	CC25IC.6
SOTBIE	DEFB	SOTBIC.6		CC25IE CC25IR	DEFB	CC251C.7
SOTBIR	DEFB	SOTBIC.7		CC25IR CC26IE	DEFB	CC26IC.6
				CC26IE CC26IR	DEFB	CC26IC.7
SSCTIE	DEFB	SSCTIC.6		CC27IE	DEFB	CC27IC.6
SSCTIR	DEFB	SSCTIC.7		002/15		

4

99/05/08 23:03:14		
CC27IR	DEFB	CC27IC.7
CC28IE	DEFB	CC28IC.6
CC28IR	DEFB	CC28IC.7
CC29IE	DEFB	CC29IC.6
CC29IR	DEFB	CC29IC.7
CC30IE	DEFB	CC30IC.6
CC30IR	DEFB	CC30IC.7 CC31IC.6
CC31IE CC31IR	DEFB DEFB	CC31IC.0
ADCIE	DEFB	ADCIC.6
ADCIR	DEFB	ADCIC.7
ADEIE	DEFB	ADEIC.6 ADEIC.7
ADEIR	DEFB	ADEIC. /
TOIE	DEFB	TOIC.6 TOIC.7
TOIR	DEFB	TIIC.6
TIIE	DEFB DEFB	T1IC.8
TIIR	DEFB	T7IC.6
T7IE T7IR	DEFB	T7IC.7
TRIE	DEFB	T8IC.6
T8IR	DEFB	T8IC.7
ADST	DEFB	ADCON.7
ADBSY	DEFB	ADCON.8
ADWR	DEFB	ADCON.9
ADCIN	DEFB	ADCON.10
ADCRQ	DEFB	ADCON.11
ILLBUS	DEFB	TFR.0
ILLINA	DEFB	TFR.1
ILLOPA	DEFB	TFR.2
PRTFLT	DEFB	TFR.3
UNDOPC	DEFB	TFR.7 TFR.13
STKUF	DEFB DEFB	TFR.13
STKOF NMI	DEFB	TFR.14
WDTIN	DEFB DEFB	WDTCON.0 WDTCON.1
WDTR	DEFB	
SOSTP	DEFB	SOCON.3
SOREN	DEFB	SOCON.4
SOPEN	DEFB	SOCON.5
SOFEN	DEFB	SOCON.6 SOCON.7
SOOEN	DEFB DEFB	SOCON. 9
SOPE SOFE	DEFB	SOCON.9
SODE	DEFB	S0CON.10
SOODD	DEFB	SOCON.12
SOBRS	DEFB	S0CON.13
SOLB	DEFB	SOCON.14
SOR	DEFB	SOCON.15
SSCHB	DEFB	SSCCON.4
SSCPH	DEFB	SSCCON.5
SSCPO	DEFB	SSCCON.6
SSCTEN	DEFB	SSCCON.8
SSCREN	DEFB	SSCCON.9
SSCPEN	DEFB	SSCCON.10
SSCBEN	DEFB	SSCCON.11
SSCBSY	DEFB	SSCCON.12
SSCMS	DEFB	SSCCON.14
SSCEN	DEFB	SSCCON.15

99/05/08

reg1	67b.def
------	---------

PWMIE

PWMIR

XP3IE

XP3IR

XP2IE XP2IR

XP1IE

XP1IR

XPOIE

XPOIR

def		
PTRO	DEFB	PWMCON0.0
PTR1	DEFB	PWMCON0.1
PTR2	DEFB	PWMCON0.2
PTR3	DEFB	PWMCON0.3
PTIO	DEFB	PWMCON0.4
PTI1	DEFB	PWMCON0.5
PTI2	DEFB	PWMCON0.6
PTI3	DEFB	PWMCON0.7
PIEO	DEFB	PWMCON0.8
PIE1	DEFB	PWMCON0.9
PIE2	DEFB	PWMCON0.10
PIE3	DEFB	PWMCON0.11
PIRO	DEFB	PWMCON0.12
PIR1	DEFB	PWMCON0.13
PIR2	DEFB	PWMCON0.14
PIR3	DEFB	PWMCON0.15
PEN0	DEFB	PWMCON1.0
PEN1	DEFB	PWMCON1.1
PEN2	DEFB	PWMCON1.2
PEN3	DEFB	PWMCON1.3
PM0	DEFB	PWMCON1.4
PM1	DEFB	PWMCON1.5
PM2	DEFB	PWMCON1.6
PM3	DEFB	PWMCON1.7
PB01	DEFB	PWMCON1.12
PS2	DEFB	PWMCON1.14
PS3	DEFB	PWMCON1.15

DEFB

PWMIC.6

PWMIC.7

XP3IC.6 XP3IC.7

XP2IC.6

XP2IC.7

XP1IC.6

XP1IC.7

XPOIC.6

XPOIC.7

B.6 42V Bus CAN Node 2

On the next page starts the code for the 42V bus CAN node 2. The files for the node are as follows.

- 1. comp242.bat
- 2. main242.asm
- 3. cnmod242.asm
- 4. canmo242.asm
- 5. cnint242.asm
- 6. atod 242.asm
- $7. \ tmrs 242.asm$
- 8. linker.lnv
- 9. Reg167b.def



al66 main242.asm al66 cnmod242.asm al66 cnint242.asm al66 cnint242.asm al66 atod242.asm al66 tmrs242.asm al66 tmrs242.asm ll66 LINK main242.obj cnmod242.obj canmo242.obj cnint242.obj atod242.obj tmrs242.obj TO locatein.lno ll66 @linker.lnv

ihex166 -i16 locate.out -o main242.hex



main242.asm



\$SEGMENTED ;; Initialize CAN Bus \$EXTEND CALL canin ; Call the CAN initialization function \$EXTSFR ;; End of CAN Bus Initialization **\$EXTSSK** ; CAN USE ALL internal RAM for Stack \$EXTMEM meto: NOP ; just loop here waiting \$NOMOD166 \$STDNAMES(reg167b.def) NOP JMP meto \$SYMBOLS RET ; return main ENDP NAME main RBANK1 COMREG R0-R15 ; define a common register area of 16 register mainseg ENDS SSKDEF 4 ; default stack size of 256 Words startupsec SECTION CODE ; codesegment that contains reset int pointer sysreset PROC TASK INTNO=0H ; reset interrupt number is zero at Oh ASSUME DPP3:SYSTEM ; forces next instruction to be located at Oh ORG 000H JMP start ; installs a pointer to the startup routine RETI EXTERN canin: FAR ; Can function ; return from interrupt ; external atod initialization EXTERN atod_initialize:FAR sysreset ENDP EXTERN atod_timer_initialize:FAR startupsec ENDS END mainseg SECTION CODE main PROC FAR start: DISWDT ; disable the watchdog timer BSET IEN ; Globally Enable Interrupts both global ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh EINIT ; end initialization ;; End of external memory bus initialization ;; Initialize the Data Page pointers for this section MOV DPP3, #03h ; make DPP3 point to system ;; End of Data Page Pointer Initialization ;; Make the direction of Port 2 to output MOV DP2, ONES ;; Make sure Port 2 is in push/pull mode MOV ODP2, ONES ;; Initialize The Stack ;; The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #OFBFFh the highest ;; byte that the stack pointers can point to is #OFBFEh MOV STKUN, #0FBFEh; Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP. #0FBFEh : Set the Stack Pointer ;; End of Stack Initialization :: Initialize the Analog to Digital Converter CALL atod initialize; atod ;; End of A/D initialization ;; Initialize A/D timer CALL atod_timer_initialize; timers ;; End of A/D timer initialization

99/05/09 11:00:32

cnmod242.asm



\$SEGMENTED \$EXTEND \$EXTSFR		RET canin ENDP	
\$EXISFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.d \$SYMBOLS	ef)	setall PROC FAR ;; by using a counter ;; objects along the v PUSH R4	; This Procedure sets all of the Mess objs invalid it counts up to 15 and initializes all of the message way.
NAME canmod		PUSH R4 PUSH R5 AND R5, ZEROS	
RBANK1 COMREG RO-R GLOBAL canin	; The function must be declared Global at the ; beginning of the module	OR R5, #01h AND R2,ZEROS OR R2,#0EF10h AND R4, ZEROS	; Set counter to 1 for first MO ; Set pointer to MO1
EXTERN canmocfg:FA	R ; configures specific Message objects	OR R4, #5555h	; Set R4 to make MObs invalid
ASSUME DPP3:SYSTEM		nextreg:MOV [R2],R4 ADD R2,#10h	; make all message objects invalid
canfunc SECTION CO	DE ; codesegment that contains reset int pointer	CMPI1 R5,#0Fh JMPA CC_NZ,nextreg	a la construcción de la cons
canin PROC FAR PUSH RO PUSH R1		POP R5 POP R4 POP R2 RET	
AND C1CSR,Z	of the CAN control registers EROS ; set control register to zero	setall ENDP	
MOV R1, #00 OR C1CSR,R1		canfunc ENDS END	
MOV R1, #03	ZEROS ; set Bit timing register to zero 447h ; set for 125k operation 1 ; set Bit timing register parameters		
MOV R1, #0F	ZEROS ; set Global Mask short register to zero FFFh ; EOFF is what DAVE initialize 1 ; set GMS		
AND ClUGML, MOV R1, #0F OR ClUGML,			
MOV R1, #0F AND C1LGML, OR C1LGML,	ZEROS		
AND C1UMLM, OR C1UMLM, AND C1LMLM,	R1 ; upper mask of last register		
OR C1LMLM,	R1 ; lower mask of last register		
CALL setall	; sets all of the CAN registers to off		
CALL canmoc			
EXTR #4	N interrupt and Initialize CAN module		
AND XPOIC, AND R0,ZERO			
OR R0,#0073 OR XP0IC,R0 AND R1, ZER	; Configure CAN interrupt Control Register		
	41h ; crashes if I clear the CPU access to the BTR		

99/05/09 10:16:58 canmo24	42.asm
<pre>\$SEGMENTED \$EXTEND \$EXTSPR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS NAME canmo RBANK1 COMREG R0-R15 ; declare bank of 16 global registers GLOBAL canmocfg</pre>	<pre>MOV [R2],R1 ; set MO3's Control register ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero OR R3, #0C003h ; The number is the Message ID for Message Object 3 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0038h ; put 000h into first data byte and set to receive MOV MCD_M3,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o f data MOV DATA_M3, ZEROS ; Fill the Data of the M0 with Zeros</pre>
<pre>can_module SECTION CODE ASSUME DPP3:SYSTEM canmoofg PROC FAR PUSH R1 PUSH R2 PUSH R3 ;; Now set specific CAN control Registers ;; initialize message object 1 ;; initializing this object to be invalid does or removing the code until ;; the comment "Setup CAN interrupt and Initialize" does ;; nothing to prevent the occurrance of the interrupt for the CAN system MOV R2, #MCR_M1 ; start of Message Object 1 AND R1, ZEROS OR R1, #5599h ; Generate a Receive Interrupt if this message object ac tivates MOV [R2],R1 ; set MO1's Control register</pre>	<pre>;; Initialize Message Object 4 MOV R2, #MCR_M4 ; start of Message Object 4 AND R1, ZEROS OR R1, #5595h ; MOV [R2],R1 ; set MO4'S Control register ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero OR R3, #00019h ; The number is the Message ID for Message Object 4 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0038h ; put 0AAh into first data byte and set to receive MOV MCD_M4,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o f data MOV DATA_M4, ZEROS ; fill the data of the MO with ZEROS</pre>
ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R3 to OR R3, #08003h ; message id = #0003h ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #0030h ; put 0AAh into first data byte and set to receive MOV MCD_M1,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes of data MOV DATA_M1, ZEROS ; fill the Data of the M0 with Zeros ;; Initialize Message Object 2 MOV R2, #MCR_M2 ; start of Message Object 2 AND R1, #5599h ; RECEIVE INTERRUPT enabled MOV [R2],R1 ; set MO2'S Control register ADD R2, #2h ; point to Upper Arbitration register ADD R2, #2h ; point to Upper Arbitration Register ADD R2, #2h ; point to the Lower Arbitration Register ADD R2, #2h ; point to the Lower Arbitration Register ADD R2, #2h ; point to the Lower Arbitration Register MOV [R2],R3 ; message id = 0 ADD R2, #2h ; point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS OR R1, #6030h ; put 000h into first data byte and set to receive MOV [R2], ZEROS ; put 000h into first data byte and set to receive MOV [R2], ZEROS ; Fill the Data of the M0 with Zeros ;; Initialize Message Object 3 MOV DATA_M2, ZEROS ; Fill the Data of the M0 with Zeros ;; Initialize Message Object 3 MOV R2, #MCR_M3 ; start of Message Object 3	<pre>;; Initialize Message Object 5 MOV R2, #MCR_M5 ; start of Message Object 5 AND R1, ZEROS OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero OR R3, #00017h ; The number is the Message ID for Message Object 5 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS or standard Message object so lowerarb = 0h AND R1, ZEROS ; put 0AAh into first data byte and set to receive MOV MCD_M5,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o f data MOV DATA_M5, ZEROS ; fill the data of the M0 with ZEROS OR R1, #0038h ; start of Message Object 6 MOV R2, #MCR_M6 ; start of Message Object 6 AND R1, ZEROS OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register ADD R2,#2h ; point to Upper Arbitration register ADD R2,#2h ; point to Upper Arbitration register ADD R3, ZEROS ; standard Message ID for Message Object 6 MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register MOT [R2], ZEROS ; standard Message object so lowerarb = 0h ADD R2, #2h ; Point to the Lower Arbitration Register MOT [R2], ZEROS ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R2, #2h ; Standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R2, #2h ; standard Message object so lowerarb = 0h ADD R1, ZEROS ; standard Message object so lowerarb = 0h ADD R2, #2h ; standard Message object so lowerarb = 0h ADD R2, #2h ; standard Message object so lowerarb = 0</pre>
MOV R2, #MCR_M3 ; start of Message Object 3 AND R1, ZEROS OR R1, #5595h ; Generate a receive interrupt if this message object ac tivates	AND R1, ZEROS OR R1, #0038h ; put 0AAh into first data byte and set to receive MOV MCD_M6,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o f data



canmo242.asm



MOV DATA_M6, ZEROS

; fill the data of the MO with ZEROS

POP R3 POP R2 POP R1 RET canmocfg ENDP

can_module ENDS END

99/05/09 11:40:10

\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS

NAME canint RBANK1 COMREG R0-R15

; declare bank of 16 global registers

ASSUME DPP3:SYSTEM

can_interrupts SECTION CODE

can_receive_interrupt PROC TASK INTNO=040h ORG 0100h CALL can_receive_interrupt_handler RETI

can_receive_interrupt ENDP

can_receive_interrupt_handler PROC FAR PUSH RO PUSH R1 PUSH R2 MOVB RLO, INTID ; Read the CAN interrupt ID buffer ; See if the interrupt came from M01 CMPB RLO, #03h JMP cc_Z, message_one_interrupt; if interrupt from M01 handle MOV R1, #05555h MOV R2, #05599h MOV MCR_M2, R1 MOV R0, DATA_M2 MOV MCR_M2, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch MOV R2, MCR_M6 MOV MCR_M6, R1 MOV DATA_M6, R0 MOV MCR_M6, R2

MOV MCR_M6, R2 CMP R0, #01h JMP cc_NZ, turn_off_heated_rear_window BSET P2.1 JMP exit_function

turn_off_heated_rear_window: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.1 JMP exit_function

message_one_interrupt:

MOV R1, #05555h MOV R2, #05599h MOV MCR_M1, R1 MOV R0, DATA_M1 MOV MCR_M1, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch

MOV R2, MCR_M5 MOV MCR_M5, R1

cnint242.asm

MOV DATA_M5, R0

MOV MCR_M5, R2 CMP R0, #01h JMP cc_NZ, turn_heater_off BSET P2.0 JMP exit_function

turn_heater_off: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.0 exit_function: MOV R2, #0EFFFh AND C1CSR, R2 POP R2 POP R1 POP R0 RET can_receive_interrupt_handler ENDP

can_interrupts ENDS END



atod242.asm



SSEGMENTED PUSH R3 PUSH R4 \$EXTEND **\$EXTSFR** PUSH MDH SEXTSSK ; CAN USE ALL internal RAM for Stack PUSH MDL \$EXTMEM MOV R2, ADDAT \$NOMOD166 \$STDNAMES(reg167b.def) MOV R0, R2 : This is so we can isolate the A/D channel from whi SYMBOLS ch the data is coming : This is so we can isolate the DATA on the A/D MOV R3, R2 AND R3, #03FFh ; this isolates the A/D data MOV R4, #01h ; No scaling on microcontroller name atod AND R0, #0F000h ; The channel information is located in the upper nibble ASSUME DPP3:SYSTEM CMP R0, #01000h ; See if the information is coming from Channel 1 of the A/ RBANK1 COMREG R0-R15 D JMP cc_Z, Rear_Seat_Heater_current GLOBAL atod_initialize ;; This A/D is set up to measure the current in two different MOV R0, #05555h ; This bit pattern deactives MCRs ;; loads. Because this software is to be used as part of MOV R1, MCR_M3 ; SAVE the Configuration of the MCR ;; 42volt bus node 1, it uses the names of the loads that MOV MCR M3, RO ; Kill the Message Control Register ;; that node is supposed to control. ;; The analog to digital converter uses Port 5 MULU R3, R4 NOP MOV DATA_M3, MDL ; for real atod_setup SECTION CODE MOV P2, R2 ; for testing purposes ; MOV MCR_M3, R1 atod initialize PROC FAR BSET T3R :: Initialize variables JMP exit routine ;; This below line of code setups up the A/D converter :: for 2 channels and single conversion. Rear Seat Heater current: ;; It is also set for "Wait for read mode" ;; so the converter will wait for the user program to read MOV R0, #05555h ; This bit pattern deactives MCRs ;; the buffer before processing the next channel. MOV R1, MCR_M4 ; SAVE the Configuration of the MCR MOV ADCON, #0A221h ; setup A/D control register MOV MCR_M4, R0 ; Kill the Message Control Register MOV R0, #04h ;test code ADD P2, R0 ;test code :: Set the channel to which the data should be written ;; when the first "A/D is done" interrupt occurs MULU R3, R4 :: The below code sets up the A/D's Interrupt control register NOP ;; The A/D is setup to have a group of 2 and a level of 10 MOV DATA_M4, MDL : for real MOV ADCIC, #006Fh MOV MCR M4, R1 RET atod initialize ENDP exit_routine: atod setup ENDS POP MDL POP MDH POP R4 atod_handlers SECTION CODE atod_handler PROC TASK INTNO=028h POP R3 ORG 0A0H POP R2 CALL atod_function POP R1 RETT POP RO RET atod handler ENDP atod function ENDP atod_function PROC FAR atod handlers ENDS ;; this function works by seeing if the converter is converting ;; for the heater_measurement. If the bit is set, then END :: the bit gets cleared and the IP jumps to where the ;; value in the converter is moved into the heater_current ;; variable. ;; otherwise the bit gets set and the value is moved into ;; the heated rear_window_current variable PUSH RO PUSH R1 PUSH R2



tmrs242.asm

\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$EXTINSTR \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS	; These are assembler controls ; Assembler controls end here
NAME timer_functions ASSUME DPP3:SYSTEM RBANK1 COMREG R0-R15	
GLOBAL atod_timer_initialize	
atod_timer SECTION CODE atod_timer_initialize PROC FAR MOV T3CON, #0004h MOV T3IC, #002Bh MOV T3, #0000h BSET T3IE BSET T3R RET atod_timer_initialize ENDP	; setup Core Timer T3 ; Make the value in the counter equal to zero ; enable the timer interrupt ; start the timer
atod_interrupt PROC TASK INTNO= ORG 08Ch CALL atod_timer_handler RETI atod_interrupt ENDP	
atod_timer_handler PROC FAR BCLR T3R BSET ADST RET atod_timer_handler ENDP atod_timer ENDS END	; stop the timer ; start an A/D conversion



LOCATE main.lno {GENERAL} IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (040000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO main.out



99/05/08 23:03:14

reg167b.def

***********	*****	*****
;** @(#)reg167h		1.10 12/18/97
;**		
;** Register de	finiti	ons for the SAB C167
;** This file o	contain	s all SFR names and BIT names
;** This file c	an be	supplied to rm166 and a166 (STDNAMES control)
; * * * * * * * * * * * * * *	*****	*******
TRUE	DEFB	OFF20h.0, RW
NODE142	DEFB	OFF20h.1, RW
0100P	DEEA	AFFOCh
CICSR	DEFA DEFA	0EF00h 0EF02h
INTID C1BTR	DEFA	0EF04h
CIGMS	DEFA	0EF06h
CIUGML	DEFA	0EF08h
C1LGML	DEFA	0EF0Ah
C1UMLM	DEFA	0EF0Ch
C1LMLM	DEFA	0EF0Eh
MCR_M1	DEFA	0EF10h
MCR_M2	DEFA	0EF20h
MCR_M3	DEFA	0EF30h
MCR_M4	DEFA	0EF40h 0EF50h
MCR_M5 MCR_M6	DEFA DEFA	0EF60h
MCR_M7	DEFA	0EF70h
MCR_M8	DEFA	0EF80h
MCR_M9	DEFA	0EF90h
MCR_MA	DEFA	OEFAOh
MCR_MB	DEFA	0EFB0h
MCR_MC	DEFA	0EFC0h
MCR_MD	DEFA	0EFD0h
MCR_ME	DEFA	OEFEOh
MCR_MF	DEFA	0EFF0h
MCD_M1	DEFA	0EF16h
MCD_M2	DEFA DEFA	0EF26h 0EF36h
MCD_M3 MCD_M4	DEFA	0EF46h
MCD_M5	DEFA	0EF56h
MCD_M6	DEFA	0EF66h
MCD_M7	DEFA	0EF76h
MCD_M8	DEFA	0EF86h
MCD_M9	DEFA	0EF96h
MCD_MA	DEFA	0EFA6h
MCD_MB	DEFA	0EFB6h
MCD_MC	DEFA	0EFC6h
MCD_MD	DEFA	0EFD6h 0EFE6h
MCD_ME DATA_M1	DEFA DEFA	0EF18h
DATA M2	DEFA	0EF28h
DATA_M3	DEFA	0EF38h
DATA_M4	DEFA	0EF48h
DATA_M5	DEFA	0EF58h
DATA_M6	DEFA	0EF68h
DATA_M7	DEFA	0EF78h
DATA_M8	DEFA	0EF88h
DATA_M9	DEFA	0EF98h
DATA_MA	DEFA DEFA	0EFA8h OFFReb
DATA_MB DATA_MC	DEFA	0EFB8h 0EFC8h
DATA_MC DATA MD	DEFA	0EFD8h
DATA_ME	DEFA	0EFE8h
DP8	DEFR	0FFD6h

P8	DEFR	0FFD4h
DP7	DEFR	0FFD2h
P7	DEFR	0FFD0h
DP6	DEFR	OFFCEh
P6	DEFR	0FFCCh
DP4	DEFR	OFFCAh
P4	DEFR	0FFC8h
DP3	DEFR	0FFC6h
P3	DEFR	0FFC4h
DP2	DEFR	0FFC2h
P2	DEFR	0FFC0h
SSCCON	DEFR	0FFB2h
SOCON	DEFR	0FFB0h
WDTCON	DEFR	OFFAEh
TFR	DEFR	OFFACh
P5	DEFR	0FFA2h
ADCON	DEFR	0FFA0h
TIIC	DEFR	0FF9Eh
TOIC	DEFR	0FF9Ch
ADEIC	DEFR	0FF9Ah
ADCIC	DEFR	0FF98h
CC15IC	DEFR	0FF96h
CC14IC	DEFR	0FF94h
CC13IC	DEFR	0FF92h
CC12IC	DEFR	0FF90h
CC11IC	DEFR	0FF8Eh
CC10IC	DEFR	0FF8Ch
CC9IC	DEFR	0FF8Ah
CC8IC	DEFR	0FF88h
CC7IC	DEFR	0FF86h
CC6IC	DEFR	0FF84h
CC5IC	DEFR	0FF82h
CC4IC	DEFR	0FF80h
	DEFR	0FF7Eh
CC3IC		0FF7Eh
CC2IC	DEFR	0FF7Ch 0FF7Ah
CC1IC	DEFR	
CCOIC	DEFR	0FF78h
SSCEIC	DEFR	0FF76h 0FF74h
SSCRIC	DEFR	
SSCTIC	DEFR	0FF72h
SOEIC	DEFR	0FF70h
SORIC	DEFR	0FF6Eh
SOTIC	DEFR	0FF6Ch
CRIC	DEFR	0FF6Ah
TGIC	DEFR	0FF68h
TSIC	DEFR	0FF66h
T4IC	DEFR	0FF64h
TJIC	DEFR	0FF62h
T2IC	DEFR	0FF60h
CCM3	DEFR	0FF58h
CCM2	DEFR	0FF56h
CCM1	DEFR	0FF54h
CCM0	DEFR	0FF52h
T01CON	DEFR	0FF50h
T6CON	DEFR	0FF48h
T5CON	DEFR	0FF46h
T4CON	DEFR	0FF44h
T3CON	DEFR	0FF42h
T2CON	DEFR	0FF40h
PWMCON1	DEFR	0FF32h
PWMCON0	DEFR	0FF30h
CCM7	DEFR	0FF28h
CCM6	DEFR	0FF26h
CCM5	DEFR	0FF24h
CCM4	DEFR	0FF22h

2

reg167b.def

99/05/0 23:03:1		
T78CON	DEFR	0FF20h
P1H	DEFR	0FF06h
P1L	DEFR	0FF04h
POH	DEFR	0FF02h
POL	DEFR	OFFOOh
PECC7	DEFR	OFECEh
PECC6	DEFR	OFECCh
PECC5	DEFR	OFECAh
PECC4	DEFR DEFR	0FEC8h 0FEC6h
PECC3 PECC2	DEFR	0FEC4h
PECC1	DEFR	0FEC2h
PECCO	DEFR	OFECOh
SRCP0	DEFA	0FCE0h
DSTP0	DEFA	0FCE2h
SRCP1	DEFA	0FCE4h
DSTP1	DEFA	0FCE6h
SRCP2	DEFA	0FCE8h
DSTP2	DEFA	OFCEAh
SRCP3	DEFA	OFCECh
DSTP3 SRCP4	DEFA	0FCEEh 0FCF0h
DSTP4	DEFA	0FCF2h
SRCP5	DEFA	0FCF4h
DSTP5	DEFA	0FCF6h
SRCP6	DEFA	0FCF8h
DSTP6	DEFA	OFCFAh
SRCP7	DEFA	0FCFCh
DSTP7	DEFA	OFCFEh
SOBG	DEFR	0FEB4h
SORBUF	DEFR	OFEB2h,
SOTBUF WDT	DEFR	OFEBOh, OFEAEh,
ADDAT	DEFR	OFEA0h
CC15	DEFR	0FE9Eh
CC14	DEFR	0FE9Ch
CC13	DEFR	0FE9Ah
CC12	DEFR	0FE98h
CC11	DEFR	0FE96h
CC10	DEFR	0FE94h
CC9	DEFR	0FE92h
CC8	DEFR	0FE90h
CC7	DEFR DEFR	0FE8Eh 0FE8Ch
CC6 CC5	DEFR	OFESCH
CC4	DEFR	0FE88h
CC3	DEFR	0FE86h
CC2	DEFR	0FE84h
CC1	DEFR	0FE82h
CC0	DEFR	0FE80h
CC31	DEFR	0FE7Eh
CC30	DEFR	0FE7Ch
CC29	DEFR	0FE7Ah
CC28	DEFR	0FE78h 0FE76h
CC27 CC26	DEFR DEFR	0FE76h
CC25	DEFR	0FE72h
CC24	DEFR	0FE70h
CC23	DEFR	0FE6Eh
CC22	DEFR	0FE6Ch
CC21	DEFR	0FE6Ah
CC20	DEFR	0FE68h
CC19	DEFR	0FE66h
CC18	DEFR	0FE64h
CC17	DEFR	0FE62h

r W r

- 1			
	CC16	DEFR	0FE60h
	T1REL	DEFR	0FE56h
	TOREL	DEFR	0FE54h
	T1	DEFR	0FE52h
	то	DEFR	0FE50h
	CAPREL	DEFR	0FE4Ah
			0FE48h
	Т6	DEFR	
	т5	DEFR	0FE46h
	т4	DEFR	0FE44h
	Т3	DEFR	0FE42h
	т2	DEFR	0FE40h
	PW3	DEFR	0FE36h
			0FE34h
	PW2	DEFR	
	PW1	DEFR	0FE32h
	PWO	DEFR	0FE30h
	1.110		
	; Extended	sfr area	
	0009	DEED	OFIDER
	ODP8	DEFR	0F1D6h
	ODP7	DEFR	0F1D2h
	ODP6	DEFR	0F1CEh
	ODP3	DEFR	0F1C6h
	PICON	DEFR	0F1C4h
	ODP2	DEFR	0F1C2h
	EXICON	DEFR	0F1C0h
	SOTBIC	DEFR	0F19Ch
	XP3IC	DEFR	0F19Eh
	XP2IC	DEFR	0F196h
	XP1IC	DEFR	0F18Eh
	XPOIC	DEFR	0F186h
	PWMIC	DEFR	0F17Eh
	TSIC	DEFR	0F17Ch
	T7IC	DEFR	0F17Ah
	CC31IC	DEFR	0F194h
	CC30IC	DEFR	0F18Ch
	CC29IC	DEFR	0F184h
	CC28IC	DEFR	0F178h
	CC27IC	DEFR	0F176h
	CC26IC	DEFR	0F174h
	CC25IC	DEFR	0F172h
	CC24IC	DEFR	0F170h
	CC23IC	DEFR	0F16Eh
	CC22IC	DEFR	0F16Ch
	CC21IC	DEFR	0F16Ah
	CC20IC	DEFR	0F168h
	CC19IC	DEFR	0F166h
	CC18IC	DEFR	0F164h
	CC17IC	DEFR	0F162h
	CC16IC	DEFR	0F160h
		DEFR	0F108h
	RPOH		
	DP1H	DEFR	0F106h
	DP1L	DEFR	0F104h
	DPOH	DEFR	0F102h
	DPOL	DEFR	0F100h
	SSCBR	DEFR	0F0B4h
	SSCRB	DEFR	0F0B2h
	SSCTB	DEFR	OFOBOh
	ADDAT2	DEFR	OFOAOh
	T8REL	DEFR	0F056h
	T7REL	DEFR	0F054h
	T8	DEFR	0F052h
	т7	DEFR	0F050h
	PP3	DEFR	0F03Eh
	PP2	DEFR	0F03Ch
	PP1	DEFR	0F03Ah
		DEFIN	or ophil

99/05/08 23:03:14

Contraction of the second s					
BB0	DEFR	0F038h	AN13	DEFB	P5.13
PPO			AN14	DEFB	P5.14
PT3	DEFR	0F036h			
PT2	DEFR	0F034h	AN15	DEFB	P5.15
PT1	DEFR	0F032h	T6EUD	LIT	'AN10'
	DEFR	OF030h	T5EUD	LIT	'AN11'
PTO	DEFR	or oson			
			TGIN	LIT	'AN12'
; Bit names			T5IN	LIT	'AN13'
CCOIO	DEFB	P2.0	T4EUD	LIT	'AN14'
			T2EUD	LIT	'AN15'
CC1IO	DEFB	P2.1	12200	DII	ANIS
CC210	DEFB	P2.2			
CC310	DEFB	P2.3	POUTO	DEFB	P7.0
CC4IO	DEFB	P2.4	POUT1	DEFB	P7.1
			POUT2	DEFB	P7.2
CC510	DEFB	P2.5			
CC6IO	DEFB	P2.6	POUT3	DEFB	P7.3
CC710	DEFB	P2.7	CC28I0	DEFB	P7.4
CC8IO	DEFB	P2.8	CC29I0	DEFB	P7.5
	DEFB	P2.9	CC30IO	DEFB	P7.6
CC910					
CC10IO	DEFB	P2.10	CC31IO	DEFB	P7.7
CC11IO	DEFB	P2.11			
CC12I0	DEFB	P2.12	CC16I0	DEFB	P8.0
	DEFB	P2.13	CC17I0	DEFB	P8.1
CC13I0				DEFB	P8.2
CC14I0	DEFB	P2.14	CC18IO		
CC15I0	DEFB	P2.15	CC19I0	DEFB	P8.3
EXOIN	LIT	, CC010,	CC20I0	DEFB	P8.4
		'CC1I0'	CC21IO	DEFB	P8.5
EX1IN	LIT				
EX2IN	LIT	'CC2IO'	CC2210	DEFB	P8.6
EX3IN	LIT	'CC3IO'	CC23I0	DEFB	P8.7
	DDDD				
TOIN	DEFB	P3.0			TO1001 0
TGOUT	DEFB	P3.1	TOM	DEFB	T01CON.3
CAPIN	DEFB	P3.2	TOR	DEFB	T01CON.6
T3OUT	DEFB	P3.3	T1M	DEFB	T01CON.11
			T1R	DEFB	T01CON.14
T3EUD	DEFB	P3.4			
T2IN	DEFB	P3.7	T7M	DEFB	T78CON.3
TJIN	DEFB	P3.6	T7R	DEFB	T78CON.6
T4IN	DEFB	P3.5	T8M	DEFB	T78CON.11
			T8R	DEFB	T78CON.14
SSDI	DEFB	P3.8	ION	DEFD	170001.14
SSDO	DEFB	P3.9			
TXD0	DEFB	P3.10	ACC0	DEFB	CCM0.3
RXD0	DEFB	P3.11	ACC1	DEFB	CCM0.7
			ACC2	DEFB	CCM0.11
SSCLK	DEFB	P3.13			
CLKOUT	DEFB	P3.15	ACC3	DEFB	CCM0.15
A16	DEFB	P4.0	ACC4	DEFB	CCM1.3
			ACC5	DEFB	CCM1.7
A17	DEFB	P4.1			
A18	DEFB	P4.2	ACC6	DEFB	CCM1.11
A19	DEFB	P4.3	ACC7	DEFB	CCM1.15
A20	DEFB	P4.4			
			ACC8	DEFB	CCM2.3
A21	DEFB	P4.5			
A22	DEFB	P4.6	ACC9	DEFB	CCM2.7
A23	DEFB	P4.7	ACC10	DEFB	CCM2.11
			ACC11	DEFB	CCM2.15
2010	DEED	P5_0			
ANO	DEFB	P5.0	10010	DEED	COM2 2
AN1	DEFB	P5.1	ACC12	DEFB	CCM3.3
AN2	DEFB	P5.2	ACC13	DEFB	CCM3.7
AN3	DEFB	P5.3	ACC14	DEFB	CCM3.11
	DEFB	P5.4	ACC15	DEFB	CCM3.15
AN4				2210	00110140
AN5	DEFB	P5.5	1000		00111 -
AN6	DEFB	P5.6	ACC16	DEFB	CCM4.3
AN7	DEFB	P5.7	ACC17	DEFB	CCM4.7
	DEFB	P5.8	ACC18	DEFB	CCM4.11
AN8					
AN9	DEFB	P5.9	ACC19	DEFB	CCM4.15
AN10	DEFB	P5.10	and the second second second		
AN11	DEFB	P5.11	ACC20	DEFB	CCM5.3
		P5.12	ACC21	DEFB	CCM5.7
AN12	DEFB	EJ.14			
			1		
			8		

99/05/08 23:03:14				
ACC22	DEFB	CCM5.11		
ACC23	DEFB	CCM5.15		
ACC24	DEFB	CCM6.3		
ACC25	DEFB	CCM6.7		
ACC26	DEFB	CCM6.11		
ACC27	DEFB	CCM6.15		
ACC28	DEFB	CCM7.3		
ACC29	DEFB	CCM7.7		
ACC30	DEFB	CCM7.11		
ACC31	DEFB	CCM7.15		
T2R	DEFB	T2CON.6		
T2UD	DEFB	T2CON.7		
T2UDE	DEFB	T2CON.8		
T3R	DEFB	T3CON.6		
T3UD	DEFB	T3CON.7		
T3UDE	DEFB	T3CON.8		
T3OE	DEFB	T3CON.9		
T3OTL	DEFB	T3CON.10		
T4R	DEFB	T4CON.6		
T4UD	DEFB	T4CON.7		
T4UDE	DEFB	T4CON.8		
T5R	DEFB	T5CON.6		
T5UD	DEFB	T5CON.7		
T5UDE	DEFB	T5CON.8		
T5CLR	DEFB	T5CON.14		
T5SC	DEFB	T5CON.15		
T6R	DEFB	T6CON.6		
T6UD	DEFB	T6CON.7		
T6UDE	DEFB	T6CON.8		
T6OE	DEFB	T6CON.9		
T6OTL	DEFB	T6CON.10		
T6SR	DEFB	T6CON.15		
T2IE T2IR T3IE T3IR T4IE T4IR T5IE T5IR T6IE T6IR	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T2IC.6 T2IC.7 T3IC.6 T3IC.7 T4IC.6 T4IC.7 T5IC.6 T5IC.7 T6IC.6 T6IC.7		
CRIE	DEFB	CRIC.6		
CRIR	DEFB	CRIC.7		
SOTIE SOTIR SORIE SOEIE SOEIE SOTBIE SOTBIE SSCTIE	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	SOTIC.6 SOTIC.7 SORIC.6 SORIC.7 SOEIC.6 SOTBIC.7 SOTBIC.6 SOTBIC.7 SSCTIC.6		
SSCTIR	DEFB	SSCTIC.7		

reg167b.def

0			
	SSCRIE	DEFB	SSCRIC.6
	SSCRIR	DEFB	SSCRIC.7
	SSCEIE	DEFB	SSCEIC.6
	SSCEIR	DEFB	SSCEIC.7
	SSCTE	LIT	'SSCTEN'
	SSCRE	LIT	'SSCREN'
	SSCPE	LIT	'SSCPEN'
	SSCBE	LIT	'SSCBEN'
	CCOIE	DEFB	CCOIC.6
	CCOIR	DEFB	CCOIC.7
	CC1IE	DEFB	CC1IC.6
	CC1IR	DEFB	CC1IC.7
	CC2IE	DEFB	CC2IC.6
	CC2IR	DEFB	CC2IC.7
	CC3IE	DEFB	CC3IC.6
	CC3IR	DEFB	CC3IC.7
	CC4IE	DEFB	CC4IC.6
	CC4IR	DEFB	CC4IC.7
	CC5IE	DEFB	CC5IC.6
	CC5IR	DEFB	CC5IC.7
	CC6IE	DEFB	CC6IC.6
	CC6IR	DEFB	CC6IC.7
	CC7IE	DEFB	CC7IC.6
	CC7IR	DEFB	CC7IC.7
	CC8IE	DEFB	CC8IC.6
	CC8IR	DEFB	CC8IC.7
	CC9IE	DEFB	CC9IC.6
	CC9IR	DEFB	CC9IC.7
	CC10IE	DEFB	CC10IC.6
	CC10IR	DEFB	CC10IC.7
	CC11IE	DEFB	CC11IC.6
	CC11IR	DEFB	CC11IC.7
			CC12IC.6
	CC12IE	DEFB	
	CC12IR	DEFB	CC12IC.7
	CC13IE	DEFB	CC13IC.6
	CC13IR	DEFB	CC13IC.7
	CC14IE	DEFB	CC14IC.6
	CC14IR	DEFB	CC14IC.7
	CC15IE	DEFB	CC15IC.6
	CC15IR	DEFB	CC15IC.7
	CC16IE	DEFB	CC16IC.6
	CC16IR	DEFB	CC16IC.7
	CC17IE	DEFB	CC17IC.6
	CC17IR	DEFB	CC17IC.7
	CC18IE	DEFB	CC18IC.6
	CC18IR	DEFB	CC18IC.7
	CC19IE	DEFB	CC19IC.6
	CC19IR	DEFB	CC19IC.7
	CC20IE	DEFB	CC20IC.6
	CC20IR	DEFB	CC20IC.7
	CC21IE	DEFB	CC21IC.6
	CC21IR	DEFB	CC21IC.7
	CC22IE	DEFB	CC22IC.6
	CC22IR	DEFB	CC22IC.7
	CC23IE	DEFB	CC23IC.6
	CC23IR	DEFB	CC23IC.7
			CC24IC.6
	CC24IE	DEFB DEFB	CC24IC.8
	CC24IR		
	CC25IE	DEFB	CC25IC.6
	CC25IR	DEFB	CC25IC.7
	CC26IE	DEFB	CC26IC.6
	CC26IR	DEFB	CC26IC.7
	CC27IE	DEFB	CC27IC.6

99/05/08 23:03:14		
CC27IR	DEFB	CC27IC.7
CC28IE	DEFB	CC28IC.6
CC28IR	DEFB	CC28IC.7
CC29IE	DEFB	CC29IC.6
CC29IR	DEFB	CC29IC.7
CC30IE	DEFB	CC30IC.6
	DEFB	CC30IC.7
CC30IR		CC31IC.6
CC31IE	DEFB	CC311C.7
CC31IR	DEFB	CC311C.7
ADCIE	DEFB	ADCIC.6
ADCIR	DEFB	ADCIC.7
ADEIE	DEFB	ADEIC.6
ADEIR	DEFB	ADEIC.7
	5555	more (
TOIE	DEFB	TOIC.6
TOIR	DEFB	TOIC.7
T1IE	DEFB	TIIC.6
T1IR	DEFB	T1IC.7
T7IE	DEFB	T7IC.6
T7IR	DEFB	T7IC.7
T8IE	DEFB	T8IC.6
T8IR	DEFB	T8IC.7
ADST	DEFB	ADCON.7
ADBSY	DEFB	ADCON.8
ADWR	DEFB	ADCON.9
ADCIN	DEFB	ADCON.10
ADCRO	DEFB	ADCON.11
ADCRQ	DEFB	
ILLBUS	DEFB	TFR.0
ILLINA	DEFB	TFR.1
ILLOPA	DEFB	TFR.2
PRTFLT	DEFB	TFR.3
UNDOPC	DEFB	TFR.7
STKUF	DEFB	TFR.13
STKOF	DEFB	TFR.14
NMI	DEFB	TFR.15
WDTIN	DEFB	WDTCON.0
WDTR	DEFB	WDTCON.1
SOSTP	DEFB	SOCON.3
SOREN	DEFB	SOCON.4
SOPEN	DEFB	SOCON.5
SOFEN	DEFB	SOCON.6
SOOEN	DEFB	SOCON.7
SOPE	DEFB	SOCON.8
SOFE	DEFB	SOCON.9
SOOE	DEFB	SOCON.10
SOODD	DEFB	S0CON.12
SOBRS	DEFB	SOCON.13
SOLB	DEFB	SOCON.14
SOR	DEFB	SOCON.15
SSCHB	DEFB	SSCCON.4
SSCPH	DEFB	SSCCON.5
SSCPO	DEFB	SSCCON.6
	DEFB	
SSCTEN		SSCCON.8
SSCREN	DEFB	SSCCON.9
SSCPEN	DEFB	SSCCON.10
SSCBEN	DEFB	SSCCON.11
SSCBSY	DEFB	SSCCON.12
SSCMS	DEFB	SSCCON.14
SSCEN	DEFB	SSCCON.15

	PTR0	DEFB	PWMCON0.0
	PTR1	DEFB	PWMCON0.1
- 15	PTR2	DEFB	PWMCON0.2
	PTR3		PWMCON0.3
	PTIO	DEFB	PWMCON0.4
I	PTI1		PWMCON0.5
I	PTI2		PWMCON0.6
25	PTI3	DEFB	
1	PIEO		PWMCON0.8
1	PIE1	DEFB	PWMCON0.9
1	PIE2		
1	PIE3		PWMCON0.11
- 2	PIRO		PWMCON0.12
	PIR1		PWMCON0.13
1	PIR2		PWMCON0.14
1	PIR3	DEFB	PWMCON0.15
- 2	PENO	DEFB	PWMCON1.0
	PEN1		PWMCON1.1
1	PEN2		PWMCON1.2
]	PEN3		PWMCON1.3
- 1	PMO		PWMCON1.4
	PM1		PWMCON1.5
1	PM2		PWMCON1.6
- 2	PM3		PWMCON1.7
1	PB01	DEFB	PWMCON1.12
1	PS2	DEFB	PWMCON1.14
1	PS3	DEFB	PWMCON1.15
			90000000000000000000000000000000000000
- 8	PWMIE	DEFB	PWMIC.6
1	PWMIR	DEFB	PWMIC.7
	XP3IE		XP3IC.6
	XP3IR	DEFB	XP3IC.7
	XP2IE	DEFB	XP2IC.6
	XP2IR	DEFB	XP2IC.7
- 1	XP1IE		XP1IC.6
	XP1IR		XP1IC.7
	XPOIE		XPOIC.6
2	XPOIR	DEFB	XPOIC.7

reg167b.def

B.7 42V Bus CAN Node 3

On the next page starts the code for the 42V bus CAN node 3. The files for the node are as follows.

- 1. comp342.bat
- 2. main 342.asm
- 3. cnmod342.asm
- 4. canmo342.asm
- $5. \ cnint 342.asm$
- 6. atod342.asm
- 7. tmrs342.asm
- 8. linker.lnv
- 9. Reg167b.def



al66 main342.asm al66 cnmod342.asm al66 canmo342.asm al66 cant342.asm al66 atod342.asm al66 tmrs342.asm ll66 tINK main342.obj cnmod342.obj canmo342.obj cnint342.obj atod342.obj tmrs342.obj TO locatein.lno ll66 dlinker.lnv ihex166 -i16 locate.out -o main.hex



main342.asm



;; Initialize CAN Bus **\$SEGMENTED** ; Call the CAN initialization function CALL canin \$EXTEND ;; End of CAN Bus Initialization \$EXTSFR : CAN USE ALL internal RAM for Stack **\$EXTSSK** meto: \$EXTMEM NOP ; just loop here waiting \$NOMOD166 NOP \$STDNAMES(reg167b.def) JMP meto \$SYMBOLS RET ; return main ENDP NAME main ; define a common register area of 16 register mainseg ENDS RBANK1 COMREG R0-R15 startupsec SECTION CODE ; codesegment that contains reset int pointer ; default stack size of 256 Words SSKDEF 4 ; reset interrupt number is zero at Oh sysreset PROC TASK INTNO=0H ; forces next instruction to be located at Oh ASSUME DPP3:SYSTEM ORG 000H ; installs a pointer to the startup routine JMP start ; return from interrupt RETT EXTERN canin: FAR ; Can function sysreset ENDP ; external atod initialization EXTERN atod initialize:FAR startupsec ENDS EXTERN atod_timer_initialize:FAR END mainseg SECTION CODE main PROC FAR : disable the watchdog timer start: DISWDT BSET IEN ; Globally Enable Interrupts both global ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh ; end initialization EINIT ;; End of external memory bus initialization ;; Initialize the Data Page pointers for this section MOV DPP3, #03h ; make DPP3 point to system ;; End of Data Page Pointer Initialization ;; Make the direction of Port 2 to output MOV DP2, ONES ;; Make sure Port 2 is in push/pull mode MOV ODP2, ONES ;; Initialize The Stack ;; The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #0FBFFh the highest ;; byte that the stack pointers can point to is #OFBFEh MOV STKUN, #0FBFEh; Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP, #0FBFEh ; Set the Stack Pointer ;; End of Stack Initialization ;; Initialize the Analog to Digital Converter CALL atod_initialize; atod ;; End of A/D initialization ;; Initialize A/D timer CALL atod_timer_initialize; timers ;; End of A/D timer initialization



cnmod342.asm



IIIOOICA	Chinou.	942.asin	
\$SEGMENTED \$EXTEND		RET canin ENDP	
\$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg16' \$SYMBOLS	7b.def)	setall PROC FAR ;; by using a counter ;; objects along the v PUSH R2 PUSH R4	; This Procedure sets all of the Mess objs invalid it counts up to 15 and initializes all of the message way.
NAME canmod		PUSH R5 AND R5, ZEROS	
RBANK1 COMREG F GLOBAL canin	; The function must be declared Global at the ; beginning of the module	OR R5, #01h AND R2,ZEROS OR R2,#0EF10h AND R4,ZEROS	; Set counter to 1 for first MO ; Set pointer to MO1
EXTERN canmocf	g:FAR ; configures specific Message objects	OR R4, #5555h	; Set R4 to make MObs invalid
ASSUME DPP3:SYS	TEM	nextreg:MOV [R2],R4 ADD R2,#10h	; make all message objects invalid
canfunc SECTION	N CODE ; codesegment that contains reset int pointer	CMPI1 R5,#0Fh JMPA CC_NZ,nextreg	;
canin PROC FAI PUSH RO PUSH R1		POP R5 POP R4 POP R2 RET	
	all of the CAN control registers SR,ZEROS ; set control register to zero	setall ENDP	
MOV R1,	#0043h ; Set IE and INIT bits R,R1 ; set control register to R1's value	canfunc ENDS END	
MOV R1,	TR, ZEROS ; set Bit timing register to zero #03447h ; set for 125k operation R, R1 ; set Bit timing register parameters		
MOV R1,	MS, ZEROS ; set Global Mask short register to zero #OFFFFh ; EOFF is what DAVE initialize S, R1 ; set GMS		
	GML, ZEROS ; set Upper global mask long to zero #OFFFFh ML, R1		
	#0F8FFh GML, ZEROS ML, R1 ; lower global mask		
OR C1UM	MLM, ZEROS		
OR C1LM	LLM, R1 ; lower mask of last register		
CALL se	tall ; sets all of the CAN registers to off		
CALL ca			
EXTR #4	p CAN interrupt and Initialize CAN module		
AND XP0 AND R0,	<pre>DIC, ZEROS ; configure CAN interrupt control Register ZEROS</pre>		
OR RO,# OR XPOI AND R1,	C,R0 ; Configure CAN interrupt Control Register		
OR R1, XOR C1C POP R1	#00041h ; crashes if I clear the CPU access to the BTR		
POP RI POP RO			

99/05/09 12:36:30	canmo34	42.asm	1
\$SEGMENTED			MOV [R2],R1 ; set MO3's Control register
\$EXTEND			ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero
\$EXTSFR \$EXTMEM			OR R3, #02004h ; The number is the Message ID for Message Object 3
\$NOMOD166			MOV [R2],R3 ; message id = 0
\$STDNAMES(reg167b.def)			ADD R2, #2h ; Point to the Lower Arbitration Register
\$SYMBOLS			MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS
NAME canmo			OR R1, #0038h ; put 000h into first data byte and set to receive
RBANK1 COMREG R0-R15	; declare bank of 16 global registers		MOV MCD_M3,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o
GLOBAL canmocfg		f data	
			MOV DATA_M3, ZEROS ; Fill the Data of the MO with Zeros
can_module SECTION CODE			;; Initialize Message Object 4
			MOV R2, #MCR_M4 ; start of Message Object 4
ASSUME DPP3:SYSTEM			AND R1, ZEROS
			OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register
canmocfg PROC FAR PUSH R1			ADD R2,#2h ; point to Upper Arbitration register
PUSH R2			AND R3, ZEROS ; set R6 to zero
PUSH R3			OR R3, #06004h ; The number is the Message ID for Message Object 4
	CAN control Registers		MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register
;; initialize message	object 1 object to be invalid does or removing the code until		MOV [R2], ZEROS ; standard Message object so lowerarb = 0h
;; the comment "Setur	o CAN interrupt and Initialize does		AND R1, ZEROS
;; nothing to prevent	t the occurrance of the interrupt for the CAN system		OR R1, #0038h ; put OAAh into first data byte and set to receive
MOV R2, #MCR_M1	; start of Message Object 1	f data	MOV MCD_M4,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o
AND R1, ZEROS OR R1, #5599h	; Generate a Receive Interrupt if this message object ac	I data	MOV DATA_M4, ZEROS ; fill the data of the MO with ZEROS
tivates			- Polete Daese Berner Land Land Land Collinse
MOV [R2],R1 ; set	t MO1's Control register		
ADD R2, #2h	; point to Upper Arbitration register		;; Initialize Message Object 5
AND R3, ZEROS	; set R3 to		MOV R2, #MCR_M5 ; start of Message Object 5
OR R3, #00004h	; message id for message object 1		AND R1, ZEROS
MOV [R2],R3 ADD R2, #2h	; message id = #0003h ; Point to the Lower Arbitration Register		OR R1, #5595h ; MOV [R2],R1 ; set MO4's Control register
MOV [R2], ZEROS	; standard Message object so lowerarb = 0h		ADD R2,#2h ; point to Upper Arbitration register
AND R1, ZEROS			AND R3, ZEROS ; set R6 to zero
OR R1, #0030h	; put OAAh into first data byte and set to receive		OR R3, #00020h ; The number is the Message ID for Message Object 5
MOV MCD_M1, R1	; Databyte(0) = 0 and Set to receive and 3 bytes of data ; fill the Data of the MO with Zeros		MOV [R2],R3 ; message id = 0 ADD R2, #2h ; Point to the Lower Arbitration Register
MOV DATA_M1, ZEROS	; IIII the Data of the Mo with Zeros		MOV [R2], ZEROS ; standard Message object so lowerarb = 0h
;; Initialize Message	e Object 2		AND R1, ZEROS
MOV R2, #MCR_M2	; start of Message Object 2		OR R1, #0038h; put 0AAh into first data byte and set to receiveMOV MCD M5,R1; Databyte(0) = 0 and Set to receive and 3 bytes o
AND R1, ZEROS OR R1, #5599h	: RECEIVE INTERRUPT enabled	f data	MOV MCD_M5,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o
	t MO2's Control register	1 ducu	MOV DATA_M5, ZEROS ; fill the data of the MO with ZEROS
ADD R2, #2h	; point to Upper Arbitration register		
AND R3, ZEROS	; set R6 to zero		
OR R3, #04004h	; The number is the Message ID for Message Object 2 ; message id = 0		;; Initialize Message Object 6 MOV R2, #MCR_M6 ; start of Message Object 6
MOV [R2],R3 ADD R2, #2h	; Message 1d = 0 ; Point to the Lower Arbitration Register		AND R1, ZEROS
MOV [R2], ZEROS	; standard Message object so lowerarb = Oh		OR R1, #5595h ;
AND R1, ZEROS			MOV [R2],R1 ; set MO4's Control register
OR R1, #0030h	; put 000h into first data byte and set to receive ; Databyte(0) = 0 and Set to receive and 3 bytes of da		ADD R2,#2h ; point to Upper Arbitration register AND R3, ZEROS ; set R6 to zero
MOV MCD_M2,R1 ta	; $\operatorname{Databyte}(0) = 0$ and set to receive and 3 bytes of da		OR R3, #0001Ah ; The number is the Message ID for Message Object 6
MOV DATA_M2, ZEROS	; Fill the Data of the MO with Zeros		MOV [R2],R3 ; message id = 0
	 Supervise Amongo Longerister Victorial musical information acchangement (Servicement) 		ADD R2, #2h ; Point to the Lower Arbitration Register
;; Initialize Messag			MOV [R2], ZEROS ; standard Message object so lowerarb = 0h
MOV R2, #MCR_M3 AND R1, ZEROS	; start of Message Object 3		AND R1, ZEROS OR R1, #0038h ; put 0AAh into first data byte and set to receive
OR R1, #5595h	; Generate a receive interrupt if this message object ac		MOV MCD_M6,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes o

f data

tivates



canmo342.asm



MOV DATA_M6, ZEROS

; fill the data of the MO with ZEROS

POP R3 POP R2 POP R1 RET canmocfg ENDP

can_module ENDS END

99/05/09 11:40:10

\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS

NAME canint RBANK1 COMREG R0-R15

; declare bank of 16 global registers

ASSUME DPP3:SYSTEM

PUSH R2

can_interrupts SECTION CODE

can_receive_interrupt PROC TASK INTNO=040h ORG 0100h CALL can_receive_interrupt_handler RETI can_receive_interrupt ENDP

can_receive_interrupt_handler PROC FAR PUSH R0 PUSH R1

MOVB RL0, INTID; Read the CAN interrupt ID bufferCMPB RL0, #03h; See if the interrupt came from M01JMP cc_Z, message_one_interrupt; if interrupt from M01 handle

MOV R1, #05555h MOV R2, #05599h MOV MCR_M2, R1 MOV R0, DATA_M2 MOV MCR_M2, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch MOV R2,MCR_M6 MOV MCR_M6, R1 MOV DATA_M6, R0 MOV MCR_M6, R2

CMP R0, #01h JMP cc_NZ, turn_off_heated_rear_window BSET P2.1 JMP exit_function

turn_off_heated_rear_window: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.1 JMP exit_function

message_one_interrupt:

MOV R1, #05555h MOV R2, #05599h MOV MCR_M1, R1 MOV R0, DATA_M1 MOV MCR_M1, R2 ;; Now setup M5 so it can respond to queries about ;; the state of the switch

MOV R2, MCR_M5 MOV MCR_M5, R1

cnint342.asm



MOV DATA_M5, R0

MOV MCR_M5, R2 CMP R0, #01h JMP cc_NZ, turn_heater_off BSET P2.0 JMP exit_function

turn_heater_off: CMP R0, #0800h JMP cc_NZ, exit_function BCLR P2.0

exit_function: MOV R2, #0EFFFh

AND C1CSR, R2 POP R2 POP R1 POP R0 RET can_receive_interrupt_handler ENDP

can_interrupts ENDS END



atod342.asm



PUSH R3 SSEGMENTED PUSH R4 SEXTEND PUSH MDH SEXTSFR PUSH MDL ; CAN USE ALL internal RAM for Stack \$EXTSSK SEXTMEM MOV R2, ADDAT \$NOMOD166 ; This is so we can isolate the A/D channel from whi \$STDNAMES(reg167b.def) MOV RO, R2 SSYMBOLS ch the data is coming ; This is so we can isolate the A/D data MOV R3, R2 AND R3, #03FFh ; This isolates the A/D data MOV R4, #01h ; No scaling on microcontroller name atod AND R0, #0F000h ; The channel information is located in the upper nibble CMP R0, #01000h ; See if the information is coming from Channel 1 of the A/ ASSUME DPP3:SYSTEM D RBANK1 COMREG R0-R15 JMP cc Z, Heated Windshield current GLOBAL atod_initialize ; This bit pattern deactives MCRs ;; This A/D is set up to measure the current in two different MOV R0, #05555h MOV R1, MCR_M3 ; SAVE the Configuration of the MCR ;; loads. Because this software is to be used as part of MOV MCR M3, R0 ; Kill the Message Control Register ;; 42volt bus node 1, it uses the names of the loads that ;; that node is supposed to control. MUL R3, R4 ;; The analog to digital converter uses Port 5 NOP ; for real MOV DATA_M3, MDL MOV P2. R2 ; for testing purposes atod setup SECTION CODE : MOV MCR M3, R1 BSET T3R atod_initialize PROC FAR JMP exit routine :: Initialize variables ;; This below line of code setups up the A/D converter Heated_Windshield_current: ;; for 2 channels and single conversion. ;; It is also set for "Wait for read mode" MOV R0, #05555h ; This bit pattern deactives MCRs ;; so the converter will wait for the user program to read ; SAVE the Configuration of the MCR ;; the buffer before processing the next channel. MOV R1, MCR M4 MOV MCR M4, R0 : Kill the Message Control Register ; setup A/D control register MOV ADCON, #0A221h MOV R0, #04h :test code ADD P2, RO ;test code ;; Set the channel to which the data should be written ;; when the first "A/D is done" interrupt occurs MUL R3, R4 NOP ;; The below code sets up the A/D's Interrupt control register MOV DATA M4, MDL ; for testing purposes ;; The A/D is setup to have a group of 2 and a level of 10 MOV MCR_M4, R1 MOV ADCIC, #006Fh exit_routine: RET POP MDL atod initialize ENDP POP MDH atod setup ENDS POP R4 POP R3 atod handlers SECTION CODE atod_handler PROC TASK INTNO=028h POP R2 POP R1 ORG 0A0H POP RO CALL atod_function RET RETT atod_function ENDP atod handler ENDP atod_handlers ENDS atod function PROC FAR ;; this function works by seeing if the converter is converting END :: for the heater measurement. If the bit is set, then :: the bit gets cleared and the IP jumps to where the ;; value in the converter is moved into the heater_current ;; variable. ;; otherwise the bit gets set and the value is moved into ;; the heated_rear_window_current variable PUSH RO PUSH R1 PUSH R2



\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$EXTINSTR \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS	; These are assembler controls ; Assembler controls end here
NAME timer_functions ASSUME DPP3:SYSTEM RBANK1 COMREG R0-R15	
GLOBAL atod_timer_initialize	
	; setup Core Timer T3 ; Make the value in the counter equal to zero ; enable the timer interrupt ; start the timer
atod_interrupt PROC TASK INTNO= ORG 08Ch CALL atod_timer_handler RETI atod_interrupt ENDP	
atod_timer_handler PROC FAR BCLR T3R BSET ADST RET atod_timer_handler ENDP atod_timer ENDS END	; stop the timer ; start an A/D conversion





LOCATE locatein.lno (GENERAL) IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (040000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO locate.out



99/05/08 23:03:14

P8 DP7

Contraction of the local division of the loc				C
. * * * * * * * * * * * * * * *	******	* * * * * * * * * * * *	**********	
;** @(#)reg167h		1.10 12/1		
;**				
;** Register de	efinitio	ons for the	e SAB C167	
			names and BIT names	
;** This file of	can be s	supplied to	o rm166 and a166 (STDNAMES control)	
;**********	*****		********	
TRUE	DEFB	0FF20h.0,		
NODE142	DEFB	0FF20h.1,	1, RW	
	0003	OPPOOL		
CICSR	DEFA	0EF00h		
INTID	DEFA	0EF02h 0EF04h		
C1BTR C1GMS	DEFA DEFA	0EF04H		
CIUGML	DEFA	0EF08h		
C1LGML	DEFA	0EF0Ah		
CIUMLM	DEFA	0EF0Ch		
C1LMLM	DEFA	0EF0Eh		
MCR_M1	DEFA	0EF10h		
MCR_M2	DEFA	0EF20h		
MCR_M3	DEFA	0EF30h		
MCR_M4	DEFA	0EF40h		
MCR_M5	DEFA	0EF50h		
MCR_M6	DEFA DEFA	0EF60h 0EF70h		
MCR_M7 MCR_M8	DEFA	0EF80h		
MCR_M9	DEFA	0EF90h		
MCR_MA	DEFA	0EFA0h		
MCR_MB	DEFA	0EFB0h		
MCR_MC	DEFA	0EFC0h		
MCR_MD	DEFA	0EFD0h		
MCR_ME	DEFA	0EFE0h		
MCR_MF	DEFA	0EFF0h		
MCD_M1	DEFA	0EF16h		
MCD_M2	DEFA	0EF26h		
MCD_M3	DEFA	0EF36h		
MCD_M4	DEFA DEFA	0EF46h 0EF56h		
MCD_M5 MCD_M6	DEFA	0EF66h		
MCD_M7	DEFA	0EF76h		
MCD_M8	DEFA	0EF86h		
MCD_M9	DEFA	0EF96h		
MCD_MA	DEFA	0EFA6h		
MCD_MB	DEFA	0EFB6h		
MCD_MC	DEFA	0EFC6h		
MCD_MD	DEFA	0EFD6h		
MCD_ME	DEFA	0EFE6h		
DATA_M1	DEFA	0EF18h		
DATA_M2 DATA_M3	DEFA DEFA	0EF28h 0EF38h		
DATA_M3 DATA_M4	DEFA	0EF48h		
DATA_M5	DEFA	0EF58h		
DATA_M6	DEFA	0EF68h		
DATA_M7	DEFA	0EF78h		
DATA_M8	DEFA	0EF88h		
DATA_M9	DEFA	0EF98h		
DATA_MA	DEFA	0EFA8h		
DATA_MB	DEFA	0EFB8h		
DATA_MC	DEFA	0EFC8h		
DATA_MD	DEFA	0EFD8h		
DATA_ME	DEFA	0EFE8h		
DP8	DEFR	0FFD6h		
100000000				

P8	DEFR	0FFD4h
DP7	DEFR	0FFD2h
P7	DEFR	0FFD0h
DP6	DEFR	OFFCEh
P6	DEFR	0FFCCh
DP4	DEFR	OFFCAh
P4	DEFR	0FFC8h
DP3	DEFR	0FFC6h
P3	DEFR	0FFC4h
DP2	DEFR	0FFC2h
P2	DEFR	0FFC0h
SSCCON	DEFR	0FFB2h
SOCON	DEFR	OFFBOh
WDTCON	DEFR	OFFAEh
TFR	DEFR	OFFACh
P5	DEFR	0FFA2h
ADCON	DEFR	OFFAOh
TIIC	DEFR	0FF9Eh
TOIC	DEFR	0FF9Ch
ADEIC	DEFR	0FF9Ah
ADCIC	DEFR	0FF98h
CC15IC	DEFR	0FF96h
CC14IC	DEFR	0FF94h
CC13IC	DEFR	0FF92h
CC12IC	DEFR	0FF90h
CC11IC	DEFR	0FF8Eh
CC10IC	DEFR	0FF8Ch
CC9IC	DEFR	0FF8Ah
CC8IC	DEFR	0FF88h
CC7IC	DEFR	0FF86h
CC6IC	DEFR	0FF84h
CC5IC	DEFR	0FF82h
CC4IC	DEFR	0FF80h
CC3IC	DEFR	0FF7Eh
CC2IC	DEFR	0FF7Ch
CC1IC	DEFR	0FF7Ah
CCOIC	DEFR	0FF78h
SSCEIC	DEFR	0FF76h
SSCRIC	DEFR	0FF74h
SSCTIC	DEFR	0FF72h
SOEIC	DEFR	0FF70h
SORIC	DEFR	0FF6Eh
SOTIC	DEFR	0FF6Ch
CRIC	DEFR	0FF6Ah
TGIC	DEFR	0FF68h
T5IC	DEFR	0FF66h
T4IC	DEFR	0FF64h
T3IC	DEFR	0FF62h
T2IC	DEFR	0FF60h
CCM3	DEFR	0FF58h
CCM2	DEFR	0FF56h
CCM1	DEFR	0FF54h
CCM0	DEFR	0FF52h
TOICON	DEFR	0FF50h
T6CON	DEFR	0FF48h
T5CON	DEFR	0FF46h
T4CON	DEFR	0FF44h
T3CON	DEFR	0FF42h
T2CON	DEFR	0FF40h
PWMCON1	DEFR	0FF32h
PWMCON0	DEFR	0FF30h
CCM7	DEFR	0FF28h
CCM6	DEFR	0FF26h
CCM5	DEFR	0FF24h
CCM4	DEFR	0FF22h

DEFR

0FFD4h

23:03	:14		
T78CON	DEFR	0FF20h	
P1H	DEFR	0FF06h	
P1L	DEFR	0FF04h	
POH	DEFR	0FF02h	
POL	DEFR	OFFOOh	
PECC7	DEFR	0FECEh	
PECC6	DEFR	0FECCh	
PECC5	DEFR	OFECAh	
PECC4	DEFR	0FEC8h	
PECC3	DEFR	0FEC6h	
PECC2	DEFR	0FEC4h	
PECC1	DEFR	0FEC2h 0FEC0h	
PECCO SRCPO	DEFR DEFA	0FCE0h	
DSTP0	DEFA	0FCE0h	
SRCP1	DEFA	0FCE2h	
DSTP1	DEFA	0FCE6h	
SRCP2	DEFA	0FCE8h	
DSTP2	DEFA	OFCEAh	
SRCP3	DEFA	OFCECh	
DSTP3	DEFA	OFCEEh	
SRCP4	DEFA	0FCF0h	
DSTP4	DEFA	0FCF2h	
SRCP5	DEFA	0FCF4h	
DSTP5	DEFA	0FCF6h	
SRCP6	DEFA	0FCF8h	
DSTP6	DEFA	OFCFAh	
SRCP7	DEFA	OFCFCh	
DSTP7	DEFA	OFCFEh	
SOBG	DEFR	0FEB4h	
SORBUF	DEFR	OFEB2h,	
SOTBUF	DEFR	OFEBOh,	
WDT	DEFR	OFEAEh,	r
ADDAT	DEFR	0FEA0h 0FE9Eh	
CC15 CC14	DEFR DEFR	0FE9Eh 0FE9Ch	
CC13	DEFR	0FE9Ah	
CC12	DEFR	0FE98h	
CC11	DEFR	0FE96h	
CC10	DEFR	0FE94h	
CC9	DEFR	0FE92h	
CC8	DEFR	0FE90h	
CC7	DEFR	0FE8Eh	
CC6	DEFR	0FE8Ch	
CC5	DEFR	0FE8Ah	
CC4	DEFR	0FE88h	
CC3	DEFR	0FE86h	
CC2	DEFR	0FE84h	
CC1	DEFR	0FE82h	
CC0	DEFR	0FE80h	
CC31	DEFR	0FE7Eh	
CC30	DEFR	0FE7Ch	
CC29	DEFR	0FE7Ah	
CC28	DEFR	OFE78h	
CC27	DEFR	0FE76h	
CC26	DEFR	0FE74h	
CC25	DEFR	0FE72h	
CC24	DEFR	0FE70h	
CC23	DEFR	0FE6Eh	

CC22

CC21

CC20

CC19

CC18

CC17

DEFR

DEFR

DEFR

DEFR

DEFR

DEFR

0FE6Ch

0FE6Ah

0FE68h

0FE66h

0FE64h

0FE62h

99/05/08

		1	
	. N		
	1 1100		
100000			

CC16	DEFR	0FE60h
TIREL	DEFR	0FE56h
TOREL	DEFR	0FE54h
T1	DEFR	0FE52h
то	DEFR	0FE50h
CAPREL	DEFR	0FE4Ah
Т6	DEFR	0FE48h
т5	DEFR	0FE46h
т4	DEFR	0FE44h
Т3	DEFR	0FE42h
т2	DEFR	0FE40h
PW3	DEFR	0FE36h
PW2	DEFR	0FE34h
	DEED	
PW1	DEFR	0FE32h
PWO	DEFR	0FE30h
; Extended sfr	area	
		0.01
ODP8	DEFR	0F1D6h
ODP7	DEFR	0F1D2h
ODP6	DEFR	0F1CEh
ODP3	DEFR	0F1C6h
PICON	DEFR	0F1C4h
ODP2	DEFR	0F1C2h
EXICON	DEFR	0F1C0h
SOTBIC	DEFR	0F19Ch
XP3IC	DEFR	0F19Eh
XP2IC	DEFR	0F196h
XP1IC	DEFR	0F18Eh
XPOIC	DEFR	0F186h
PWMIC	DEFR	0F17Eh
TRIC	DEFR	0F17Ch
T7IC	DEFR	0F17Ah
CC31IC	DEFR	0F194h
CC30IC	DEFR	0F18Ch
CC29IC	DEFR	0F184h
CC28IC	DEFR	0F178h
CC27IC	DEFR	0F176h
CC26IC	DEFR	0F174h
CC25IC	DEFR	0F172h
CC24IC	DEFR	0F170h
CC23IC	DEFR	0F16Eh
CC22IC	DEFR	0F16Ch
CC21IC	DEFR	0F16Ah
CC20IC	DEFR	0F168h
CC19IC	DEFR	0F166h
CC18IC	DEFR	0F164h
CC17IC	DEFR	0F162h
		0F160h
CC16IC	DEFR	
RPOH	DEFR	0F108h
DP1H	DEFR	0F106h
DPIH	DEFR	
DP1L	DEFR	0F104h
	DEFR	0F102h
DPOH		
DPOL	DEFR	0F100h
SSCBR	DEFR	0F0B4h
SSCRB	DEFR	0F0B2h
SSCTB	DEFR	0F0B0h
ADDAT2	DEFR	OFOAOh
T8REL	DEFR	0F056h
T7REL	DEFR	0F054h
т8	DEFR	0F052h
т7	DEFR	0F050h
PP3	DEFR	0F03Eh
PP2	DEFR	0F03Ch
PP1	DEFR	0F03Ah

23:03:14						
PPO	DEFR	0F038h				
PT3	DEFR	0F036h				
PT2	DEFR	0F034h				
PT1	DEFR DEFR	0F032h 0F030h				
PT0	DEFR	0103011				
; Bit names CC0IO	DEFB	P2.0				
CC1IO	DEFB	P2.1				
CC2IO	DEFB	P2.2				
CC310	DEFB	P2.3				
CC410 CC510	DEFB DEFB	P2.4 P2.5				
CC6IO	DEFB	P2.5				
CC710	DEFB	P2.7				
CC810	DEFB	P2.8				
CC910	DEFB	P2.9				
CC10I0	DEFB DEFB	P2.10 P2.11				
CC11IO CC12IO	DEFB	P2.11 P2.12				
CC13I0	DEFB	P2.13				
CC14I0	DEFB	P2.14				
CC15I0	DEFB	P2.15				
EXOIN	LIT LIT	'CC0I0' 'CC1I0'				
EX1IN EX2IN	LIT	'CC2IO'				
EX3IN	LIT	'CC3I0'				
TOIN	DEFB	P3.0				
TGOUT	DEFB	P3.1 P3.2				
CAPIN T3OUT	DEFB DEFB	P3.2 P3.3				
T3EUD	DEFB	P3.4				
T2IN	DEFB	P3.7				
T3IN	DEFB	P3.6				
T4IN	DEFB	P3.5 P3.8				
SSDI SSDO	DEFB	P3.8 P3.9				
TXD0	DEFB	P3.10				
RXD0	DEFB	P3.11				
SSCLK	DEFB	P3.13				
CLKOUT	DEFB	P3.15				
A16	DEFB DEFB	P4.0 P4.1				
A17 A18	DEFB	P4.1 P4.2				
A19	DEFB	P4.3				
A20	DEFB	P4.4				
A21	DEFB	P4.5				
A22 A23	DEFB DEFB	P4.6 P4.7				
ANO	DEFB	P5.0 P5.1				
AN1 AN2	DEFB DEFB	P5.2				
AN3	DEFB	P5.3				
AN4	DEFB	P5.4				
AN5	DEFB	P5.5				
ANG	DEFB	P5.6 P5.7				
AN7 AN8	DEFB DEFB	P5.7 P5.8				
ANS AN9	DEFB	P5.9				
AN10	DEFB	P5.10				
AN11	DEFB	P5.11				
AN12	DEFB	P5.12				

99/05/08

def		
AN13 AN14 AN15 T6EUD T5EUD T6IN T5IN T4EUD T2EUD	DEFB DEFB LIT LIT LIT LIT LIT LIT	P5.13 P5.14 P5.15 'AN10' 'AN11' 'AN12' 'AN13' 'AN14' 'AN15'
POUT0 POUT1 POUT2 POUT3 CC28I0 CC29I0 CC30I0 CC31I0	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P7.0 P7.1 P7.2 P7.3 P7.4 P7.5 P7.6 P7.7
CC16IO CC17IO CC18IO CC19IO CC20IO CC21IO CC22IO CC23IO	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P8.0 P8.1 P8.2 P8.3 P8.4 P8.5 P8.6 P8.7
T0M T0R T1M T1R T7M T7R T8M T8R	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T01CON.3 T01CON.6 T01CON.11 T01CON.14 T78CON.3 T78CON.6 T78CON.11 T78CON.14
ACC0	DEFB	CCM0.3
ACC1	DEFB	CCM0.7
ACC2	DEFB	CCM0.11
ACC3	DEFB	CCM0.15
ACC4	DEFB	CCM1.3
ACC5	DEFB	CCM1.7
ACC6	DEFB	CCM1.11
ACC7	DEFB	CCM1.15
ACC8	DEFB	CCM2.3
ACC9	DEFB	CCM2.7
ACC10	DEFB	CCM2.11
ACC11	DEFB	CCM2.15
ACC12	DEFB	CCM3.3
ACC13	DEFB	CCM3.7
ACC14	DEFB	CCM3.11
ACC15	DEFB	CCM3.15
ACC16	DEFB	CCM4.3
ACC17	DEFB	CCM4.7
ACC18	DEFB	CCM4.11
ACC19	DEFB	CCM4.15
ACC20	DEFB	CCM5.3
ACC21	DEFB	CCM5.7

99/05/ 23:03:	
ACC22	DEFB
ACC23	DEFB
ACC24	DEFB
ACC25	DEFB
ACC26	DEFB
ACC27	DEFB
ACC28	DEFB
ACC29	DEFB
ACC30	DEFB
ACC31	DEFB
T2R	DEFB
T2UD	DEFB
T2UDE	DEFB
T3R	DEFB
TJUD	DEFB
TJUDE	DEFB
T3OE	DEFB
T3OTL	DEFB
T4R	DEFB
T4UD	DEFB
T4UDE	DEFB
T5R	DEFB

States of the state of the stat	kinon ser				
			SSCRIE	DEFB	SSCRIC.6
ACC22	DEFB	CCM5.11			
ACC23	DEFB	CCM5.15	SSCRIR	DEFB	SSCRIC.7
			SSCEIE	DEFB	SSCEIC.6
			SSCEIR	DEFB	SSCEIC.7
ACC24	DEFB	CCM6.3			
ACC25	DEFB	CCM6.7	SSCTE	LIT	'SSCTEN'
ACC26	DEFB	CCM6.11	SSCRE	LIT	'SSCREN'
			SSCPE	LIT	'SSCPEN'
ACC27	DEFB	CCM6.15			
			SSCBE	LIT	'SSCBEN'
10000	DEFB	CCM7.3			
ACC28					
ACC29	DEFB	CCM7.7			100000000000000000000000000000000000000
ACC30	DEFB	CCM7.11	CCOIE	DEFB	CCOIC.6
			CCOIR	DEFB	CCOIC.7
ACC31	DEFB	CCM7.15			CC1IC.6
			CC1IE	DEFB	
T2R	DEFB	T2CON.6	CC1IR	DEFB	CC1IC.7
		2.8 A 5 G W W B A 5 W B A	CC2IE	DEFB	CC2IC.6
T2UD	DEFB	T2CON.7			
T2UDE	DEFB	T2CON.8	CC2IR	DEFB	CC2IC.7
		10125-0-00 (1220) (1220)	CC3IE	DEFB	CC3IC.6
	DDDD		CC3IR	DEFB	CC3IC.7
T3R	DEFB	T3CON.6			
TJUD	DEFB	T3CON.7	CC4IE	DEFB	CC4IC.6
T3UDE	DEFB	T3CON.8	CC4IR	DEFB	CC4IC.7
			CC5IE	DEFB	CC5IC.6
T3OE	DEFB	T3CON.9			
TJOTL	DEFB	T3CON.10	CC5IR	DEFB	CC5IC.7
			CC6IE	DEFB	CC6IC.6
		T (2001) (CC6IR	DEFB	CC6IC.7
T4R	DEFB	T4CON.6			
T4UD	DEFB	T4CON.7	CC7IE	DEFB	CC7IC.6
T4UDE	DEFB	T4CON.8	CC7IR	DEFB	CC7IC.7
140DE	DEFD	THEOR. 0	CC8IE	DEFB	CC8IC.6
T5R	DEFB	T5CON.6	CC8IR	DEFB	CC8IC.7
TSUD	DEFB	T5CON.7	CC91E	DEFB	CC9IC.6
			CC9IR	DEFB	CC9IC.7
T5UDE	DEFB	T5CON.8			
T5CLR	DEFB	T5CON.14	CC10IE	DEFB	CC10IC.6
T5SC	DEFB	T5CON.15	CC10IR	DEFB	CC10IC.7
1550	DEFB	13CON.13	CC11IE	DEFB	CC11IC.6
T6R	DEFB	T6CON.6	CC11IR	DEFB	CC11IC.7
	DEFB	T6CON.7	CC12IE	DEFB	CC12IC.6
TGUD					CC12IC.7
T6UDE	DEFB	T6CON.8	CC12IR	DEFB	
TGOE	DEFB	T6CON.9	CC13IE	DEFB	CC13IC.6
		T6CON.10	CC13IR	DEFB	CC13IC.7
TGOTL	DEFB				
T6SR	DEFB	T6CON.15	CC14IE	DEFB	CC14IC.6
			CC14IR	DEFB	CC14IC.7
	0000	mo to	CC15IE	DEFB	CC15IC.6
T2IE	DEFB	T2IC.6			
T2IR	DEFB	T2IC.7	CC15IR	DEFB	CC15IC.7
T3IE	DEFB	T3IC.6	CC16IE	DEFB	CC16IC.6
			CC16IR	DEFB	CC16IC.7
T3IR	DEFB	T3IC.7			
T4IE	DEFB	T4IC.6	CC17IE	DEFB	CC17IC.6
T4IR	DEFB	T4IC.7	CC17IR	DEFB	CC17IC.7
			CC18IE	DEFB	CC18IC.6
T5IE	DEFB	T5IC.6			
T5IR	DEFB	T5IC.7	CC18IR	DEFB	CC18IC.7
T6IE	DEFB	TGIC.6	CC19IE	DEFB	CC19IC.6
			CC19IR	DEFB	CC19IC.7
T6IR	DEFB	TGIC.7			
			CC20IE	DEFB	CC20IC.6
CRIE	DEFB	CRIC.6	CC20IR	DEFB	CC20IC.7
			CC21IE	DEFB	CC21IC.6
CRIR	DEFB	CRIC.7			
			CC21IR	DEFB	CC21IC.7
COMTE	DEFB	SOTIC.6	CC22IE	DEFB	CC22IC.6
SOTIE			CC22IR	DEFB	CC22IC.7
SOTIR	DEFB	SOTIC.7			
SORIE	DEFB	SORIC.6	CC23IE	DEFB	CC23IC.6
	DEFB	SORIC.7	CC23IR	DEFB	CC23IC.7
SORIR			CC24IE	DEFB	CC24IC.6
SOEIE	DEFB	SOEIC.6			
SOEIR	DEFB	SOEIC.7	CC24IR	DEFB	CC24IC.7
SOTBIE	DEFB	SOTBIC.6	CC25IE	DEFB	CC25IC.6
			CC25IR	DEFB	CC25IC.7
SOTBIR	DEFB	SOTBIC.7			
			CC26IE	DEFB	CC26IC.6
CCOUTE	DEED	SSCTIC.6	CC26IR	DEFB	CC26IC.7
SSCTIE	DEFB		CC27IE	DEFB	CC27IC.6
SSCTIR	DEFB	SSCTIC.7	CC2/IE	DEFD	CC2/1C.0
			1		

99/05/08 23:03:14		
0003770	DEFB	CC27IC.7
CC27IR	DEFB	CC28IC.6
CC28IE		CC281C.8
CC28IR	DEFB	
CC29IE	DEFB	CC29IC.6
CC29IR	DEFB	CC29IC.7
CC30IE	DEFB	CC30IC.6
CC30IR	DEFB	CC30IC.7
CC31IE	DEFB	CC31IC.6
CC31IR	DEFB	CC31IC.7
ADCIE	DEFB	ADCIC.6
ADCIR	DEFB	ADCIC.7
ADEIE	DEFB	ADEIC.6
ADEIR	DEFB	ADEIC.7
TOIE	DEFB	TOIC.6
TOIR	DEFB	TOIC.7
TIIE	DEFB	T1IC.6
T1IR	DEFB	T1IC.7
T7IE	DEFB	T7IC.6
T7IR	DEFB	T7IC.7
TSIE	DEFB	T8IC.6
T8IR	DEFB	T8IC.7
ADST	DEFB	ADCON.7
ADBSY	DEFB	ADCON.8
ADWR	DEFB	ADCON.9
ADCIN	DEFB	ADCON.10
ADCRQ	DEFB	ADCON.11
ILLBUS	DEFB	TFR.0
ILLINA	DEFB	TFR.1
ILLOPA	DEFB	TFR.2
PRTFLT	DEFB	TFR.3
UNDOPC	DEFB	TFR.7
STKUF	DEFB	TFR.13
STKOF	DEFB	TFR.14
NMI	DEFB	TFR.15
WDTIN WDTR	DEFB	WDTCON.0 WDTCON.1
WDIK		
SOSTP	DEFB	SOCON.3
SOREN	DEFB	SOCON.4
SOPEN	DEFB	SOCON.5
SOFEN	DEFB	SOCON.6
SOOEN	DEFB	SOCON.7
SOPE	DEFB	SOCON.8
SOFE	DEFB	SOCON.9
SOOE	DEFB	SOCON.10
SOODD	DEFB	SOCON.12
SOBRS	DEFB	S0CON.13
SOLB	DEFB	S0CON.14
		S0CON.14
SOR	DEFB	SUCON.15
SSCHB	DEFB	SSCCON.4
SSCPH	DEFB	SSCCON.5
SSCPO	DEFB	SSCCON.6
SSCTEN	DEFB	SSCCON.8
SSCREN	DEFB	SSCCON.9
SSCPEN	DEFB	SSCCON.10
SSCBEN	DEFB	SSCCON.11
	DEFB	SSCCON.11
SSCBSY		SSCCON.12 SSCCON.14
SSCMS	DEFB	
SSCEN	DEFB	SSCCON.15

reg167b.def

PTR0	DEFB	PWMCON0.0
PTR1	DEFB	PWMCON0.1
PTR2	DEFB	PWMCON0.2
PTR3		PWMCON0.3
PTIO	DEFB	PWMCON0.4
PTI1	DEFB	PWMCON0.5
PTI2	DEFB	PWMCON0.6
PTI3	DEFB	PWMCON0.7
PIEO		PWMCON0.8
PIE1	DEFB	PWMCON0.9
PIE2	DEFB	PWMCON0.10
PIE3	DEFB	PWMCON0.11
PIRO	DEFB	
PIR1	DEFB	PWMCON0.13
PIR2	DEFB	PWMCON0.14
PIR3	DEFB	PWMCON0.15
PENO		PWMCON1.0
PEN1		PWMCON1.1
PEN2	DEFB	PWMCON1.2
PEN3	DEFB	PWMCON1.3
PM0	DEFB	PWMCON1.4
PM1	DEFB	PWMCON1.5
PM2	DEFB	PWMCON1.6
PM3	DEFB	PWMCON1.7
PB01	DEFB	PWMCON1.12
PS2	DEFB	PWMCON1.14
PS3	DEFB	PWMCON1.15
PWMIE	DEFB	
PWMIR	DEFB	PWMIC.7
XP3IE	DEFB	XP3IC.6
XP3IR	DEFB	
XP2IE	DEFB	XP2IC.6
XP2IR	DEFB	XP2IC.7
XP1IE	DEFB	XP1IC.6
XP1IR	DEFB	XP1IC.7
XPOIE	DEFB	XPOIC.6
XPOIR	DEFB	XPOIC.7

B.8 CAN Router

On the next page starts the code for the CAN Router. The files for the node are as follows.

- 1. comp.bat
- 2. main.asm
- 3. serialApril.asm
- 4. cnmod.asm
- 5. canmo.asm
- 6. canint.asm
- 7. timers.asm
- 8. linker.lnv
- 9. Reg167b.def



comp.bat

al66 main.asm al66 serialApril.asm al66 timers.asm

a166 canmod.asm

al66 canmo.asm

a166 canint.asm

1166 LINK main.obj timers.obj serialApril.obj canint.obj canmod.obj canmo.obj TO main.ln o

1166 @linker.lnv ihex166 -i16 main.out -o main.hex



main.asm



JMP meto SSEGMENTED ; return RET SEXTEND main ENDP SEXTSFR mainseg ENDS ; CAN USE ALL internal RAM for Stack SEXTSSK SEXTMEM ; codesegment that contains reset int pointer startupsec SECTION CODE \$NOMOD166 ; reset interrupt number is zero at Oh sysreset PROC TASK INTNO=0H \$STDNAMES(reg167b.def) ; forces next instruction to be located at Oh ORG 000H \$SYMBOLS ; installs a pointer to the startup routine JMP start ; return from interrupt RETI NAME main sysreset ENDP RBANK1 COMREG R0-R15 ; define a common register area of 16 register startupsec ENDS END ; default stack size of 512 Words SSKDEF 4 ASSUME DPP3:SYSTEM EXTERN serial init: FAR ; Can function EXTERN canin:FAR EXTERN serial_timer_initialize:FAR; serial mainseg SECTION CODE main PROC FAR ; disable the watchdog timer start: DISWDT ; Globally Enable Interrupts both global BSET IEN ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh ; end initialization EINIT ;; End of external memory bus initialization ;; Initialize the Data Page pointers for this section ; make DPP3 point to system MOV DPP3, #03h ;; End of Data Page Pointer Initialization ;; Initialize The Stack ;; The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #OFBFFh the highest ;; byte that the stack pointers can point to is #0FBFEh MOV STKUN, #0FBFEh; Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP, #OFBFEh ; Set the Stack Pointer ;; End of Stack Initialization MOV DP2, ONES NOP MOV P2, ZEROS ;; Initialize the Serial Port CALL serial_init ;; End of Serial Port Initialization ;; Initialize the serial port timer CALL serial_timer_initialize; pain in the ass ;; Initialize CAN Bus ; Call the CAN initialization function CALL canin ;; End of CAN Bus Initialization meto: NOP ; just loop here waiting NOP



serial~3.asm

SSEGMENTED ; These are assembler controls SEXTEND \$EXTSFR SEXTMEM SEXTINSTR \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS : Assembler controls end here NAME serial_functions ; Every File needs one of these ;; Declare a common register bank ;; This register bank is common to all files ;; which declare that they are going to use it. RBANK1 COMREG R0-R15 ;:Declare 'serial_init' global so other files can call it. GLOBAL serial init GLOBAL byte_counter GLOBAL confirm message GLOBAL message_transmitting GLOBAL message to transmit EXTERN CAN message BYTES: BYTE :: GLOBAL serial transmit in use ;; GLOBAL serial_transmit_requested ;; Assign the DPPs with the assume directive ;; this really doesn't do anything worth mentioning ;; nothing that I understand anyhow. ASSUME DPP0:incoming_message, DPP1:transmit_structure, DPP3:SYSTEM ;; Declare the Data sections to be used by the ;; serial port. incoming_message SECTION DATA BYTE GLOBAL 'RAM' start_of_received_message label BYTE ; For Looping later start_of_frame DSB 1 number of bytes DSB 1 ; length of CAN message direction_of_transmission DSB 1 message_id DSB 2 message_data DSB 8 check sum DSB 2 end_of_frame DSB 1 ; j311 byte counter DSW 1 incoming message ENDS transmit structure SECTION DATA BYTE GLOBAL 'RAM' transmit data DSB 16 receive buffer DSB 16 transmit counter DSW 1 message_to_transmit DSW 1 message transmitting DSW 1 transmit structure ENDS serial constants SECTION DATA BYTE GLOBAL 'ROM' resend_message DB '&!!Send Over!!&' time_out_message DB '&!!Time Out!!&' message length DB 16 data_structure_size DB 12 serial_constants ENDS ;; Start of the serial section code. There are X functions in ;; 3 different sections this file. :: In the 'serial_start' section there is

;; 'rechandler', 'receive_message' serial start SECTION CODE serial init PROC FAR PUSH DPP0 PUSH DPP1 PUSH DPP2 ;; Initialize the Serial Port MOV DPP0, #PAG incoming_message MOV DPP1, #PAG transmit_structure ; hjhjh AND DPP0:byte_counter, ZEROS AND DPP1:transmit_counter, ZEROS; jasdf AND DPP1:message to transmit, ZEROS; Clear the message to transmit AND DPP1:message_transmitting, ZEROS; CLEAR MESSAGE_TRANSMITTING MOV SOCON. #08011h :Sets the serial port MOV S0BG, #0040h ;Sets the baud rate to 9600 MOVB SORIC, #030h ;Sets the interrupt for the receive side MOV SOTBUF, ZEROS EXTR #1 ; enables access to ESFR for 1 command only MOVB SOTBIC, #020h ;Sets the interrupt handler for send buffer BSET SORIC.6 ;enable the receive interrupt handler ; Enables access to ESFR EXTR #1 BCLR SOTBIC.6 enable the send buffer interrupt handler MOV DP3. ONES ;set the port direction to output MOV P3, ONES ;set the outputs to 1 BCLR DP3.11 ;Set the pin direction to input BCLR P3.11 ;Not a clue ;; End of serial port initialization POP DPP2 POP DPP1 POP DPP0 RET serial init ENDP serial start ENDS serial receive SECTION CODE receive handler PROC TASK INTNO=02BH ORG OACh CALL rechandler RETI receive_handler ENDP rechandler PROC FAR ;; The first part of this procedure makes sure that ;; the byte_counter which is the offset from the start :: of the data array which is used to hold the data message is :: set to the correct value PUSH RO PUSH R1 PUSH R2 PUSH DPP0 PUSH DPP1 MOV DPPO, #PAG start_of_received_message MOV DPP1, #PAG message length MOV R0, #DPP0:start_of_received_message; me BCLR T5CON.6 ; start the timer MOV T5, #0001h : set the timer to 1 MOV R2, DPP0:byte_counter ADD RO, R2 ; me i MOVB [R0] , SORBUF ADDB RL2, #01h MOV DPP0:byte_counter, R2







:: The structure is 14 bytes long so the comparison is ;; done against #0Ch. CMPB RL2, DPP1:message_length ; know when to call the handling function JMPA cc Z, handle_message ; need to decode the message BSET T5CON.6 ; jkj ; exit function JMP receive_end handle_message: BCLR T5CON.6 ; TURN OFF THE TIMER MOV T5, #001h MOV DPP0:byte_counter, ZEROS CALL receive_message; j receive_end: POP DPP1 POP DPP0 POP R2 POP R1 POP RO RET rechandler ENDP receive_message PROC FAR PUSH DPP0 MOV DPPO, #PAG transmit_structure CALL test_checksum ; necessary ; setup and execute the CAN Message Object CALL do_the_CAN_JAZZ CALL remove_from_receive_buffer; jkj ; CMP ZEROS, DPP0:message_transmitting; jkj ; JMP cc NZ, exit_receive_message; jkj ; CALL confirm_message ; Necessary ; exit_receive_message: POP DPP0 RET receive_message ENDP remove from receive buffer PROC FAR PUSH RO PUSH R1 PUSH R2 PUSH DPP0 PUSH DPP1 PUSH DPP2 MOV DPP0, #PAG start_of_received_message MOV DPP1, #PAG transmit_structure MOV DPP2, #PAG serial_constants AND R2. ZEROS MOV R0, #DPP0:start of received message MOV R1, #DPP1:receive_buffer move received data: MOVB [R1], [R0] ADD R2, #01h ADD RO, #01h ADD R1, #01h CMPB RL2, DPP2:message_length JMP cc_NZ, move_received_data POP DPP2 POP DPP1 POP DPP0

POP R2 POP R1 POP RO RET remove from receive buffer ENDP serial receive ENDS checksum_test_functions SECTION CODE test checksum PROC FAR ; To be used as a pointer to the message PUSH RO PUSH R1 ; To be used as an accumulator PUSH R2 ; To be used to contain data structure size ; To be used as a counter PUSH R3 ; To be used for byte to word conversions PUSH R4 PUSH R5 PUSH DPP0 PUSH DPP1 PUSH DPP2 MOV DPP0, #PAG start_of_received_message; DPP0= message_id's page MOV DPP1, #PAG data_structure_size MOV DPP2, #PAG transmit_structure AND R1. ZEROS : Make the accumulator value = Zero AND R3, ZEROS ; Set the loop counter to zero ; Make R4 all zeros AND R4, ZEROS MOV R0, #DPP0:number_of_bytes; beginning of important data calculate_total: ; Loop through the entire data structure MOVB RL4, [R0+] ; Byte to word conversion done here ADD R1, R4 ; increment the loop count ADDB RL3, #01h CMPB RL3, DPP1:data_structure_size ; Cmp R3 to the size of the loop JMP cc_NZ, calculate_total; If not equal then add again MOVB RH2, DPP0:check sum MOVB RL2, DPP0:check_sum + 1 CMP R1, R2 ; computed vs received checksums JMP cc_NZ, checksum_error MOV R5, #01h ADD DPP2:message_to_transmit, R5; Indicates good reply JMP exit_checksum checksum_error: ; indicates checksum error MOV R0, #02h ADD DPP2:message_to_transmit, R0 exit checksum: POP DPP2 POP DPP1 POP DPP0 POP R5 POP R4 POP R3 POP R2 POP R1 POP RO RET test_checksum ENDP checksum_test_functions ENDS serial_transmit SECTION CODE confirm_message PROC FAR PUSH RO PUSH R1 PUSH R2

99/06/14 05:50:36

PUSH R3 PUSH R4

PUSH R5

PUSH DPP0

PUSH DPP1 PUSH DPP2

MOV R3, #01h

MOV R4, R3

AND R3, #01h

CMP R3, #01h

SUB R3, #01h

MOV R3, R4

AND R3, #02h

CMP R3, #02h

SUB R3, #02h

MOV R3, R4

AND R3, #04h CMP R3, #04h

SUB R3, #04h

MOV R3, R4

AND R3, #08h

CMP R3, #08h

JMP exit_quickly

JMP setup pointers

JMP exit_quickly

JMP setup pointers

JMP setup pointers

JMP setup_pointers

NOP

NOP

next possibility1:

next_possibility2:

next_possibility3:

NOP





MOV R0, #DPP2:CAN_message_BYTES; set R0 to point to address of CAN return me ssage SUB R3, #08h MOV DPP1:message to transmit, R3 JMP setup pointers ;; First thing to do is copy all data into the transmit next_possibility4: MOV DPP1:message_to_transmit, ZEROS ;; data data-structure ;MOV R0, #DPP1:receive_buffer; jkj ;; load DPP0 and DPP1 with the data pages of the two data structures MOV DPP2, #PAG CAN_message_BYTES; actual possibility MOV DPPO, #PAG start_of_received_message; old version MOV DPP1, #PAG transmit_structure NOP MOV R0, #DPP2: CAN message BYTES; set R0 to point to address of CAN return me MOV DPP2, #PAG serial_constants ssage ; Random NOP MOV DPP1:message_transmitting, R3 setup_pointers: MOV R0, #DPP0:start_of_received_message; test purposes ;; determine which message to transmit MOV R0, #DPP1:receive buffer; test code MOV DPP2, #PAG CAN_message_BYTES; test code ; Another RANDOM NOP ; test code MOV R3, DPP1:message_to_transmit ; Move into R3 the message to transmit NOP MOV R0, #DPP2:CAN_message_BYTES; test code ; Copy for fast recovery MOV R1, #DPP1:transmit_data ; Test code ; set the counter to zero AND R2, ZEROS ;; move the start addresses of the two data structures ;; into registers which are to be used as pointers to MOV DPP1:message_to_transmit, ZEROS ;; the data structures move data: ; Isolate possible good message MOVB [R1], [R0] ; move data from message buffer to transmit buffer ; See if good message ADD R2, #01h : Increment everyone by #01h JMP cc_NZ, next_possibility1 ADD RO, #01h MOV R0, #DPP1:receive_buffer ADD R1, #01h CMPB RL2, DPP2:message_length ;Check all data has been transfered MOV DPP1:message_to_transmit, R3 JMP cc_NZ, move_data ; if more data to transfer then loop ;; The EXTR #1 instruction allows the BSET instruction :: to access the Extended Special Function Register area. :: without the EXTR #1 instruction, there is no way you can ; Refresh R3 buffer :: access the SOTBIC register. You also need the \$EXTSFR and ; Isolate Possible Send Over :: the SEXTINSTR assembler controls (located at the top of ; See if Send Over exists :: the file) for this to work. JMP cc NZ, next_possibility2 EXTR #1 ; test only BSET SOTBIC.6 MOV R0, #DPP1:receive_buffer; test code MOV R0, #DPP2:resend_message; jkj MOV DPP1:transmit_counter, ZEROS MOV DPP1:message_to_transmit, R3 ;; Calling a TRAP is a software way of creating an interrupt ;; in this case we are causing the interrupt handler for the :: serial transmit buffer to occur. The difference between ;; calling a trap and having the interrupt be generated from :: a hardware event is that when calling a trap, the CPU :: does not change priority level TRAP #047h ; asdf JMP cc_NZ, next_possibility3 CALL transmit buffer function; Test Code : ; test only exit_quickly: MOV R0, #DPP2:time out message; actual possibility MOV R0, #DPP1:receive_buffer; test code POP DPP2 POP DPP1 POP DPPO MOV DPP1:message to transmit, R3 POP R5 POP R4 POP R3 POP R2 POP R1 POP RO RET JMP cc_NZ, next_possibility4 MOV R0, #DPP1:receive buffer; Test Code confirm_message ENDP MOV DPP2, #PAG CAN message BYTES; actual possibility transmit handler PROC TASK INTNO=047h







:: This is the interrupt handler for the Serial Transmit Buffer ;; Interrupt. It is activated when data is transmitted from ;; the transmit buffer to the transmit shift register. ORG 011Ch CALL transmit_buffer_function RETI transmit handler ENDP transmit buffer function PROC FAR PUSH RO PUSH R1 PUSH R2 PUSH R3 PUSH DPP1 PUSH DPP2 ;; make data page on have the page number for transmit_data MOV DPP1, #PAG transmit_data MOV DPP2, #PAG message_length ;; the following is curious. It moves the address of transmit_data ;; into R0, but R0 is 16 bits and the address of transmit_data is ;; actually 24 bits...must be some assembler magic going on in the ;; background MOV R0, #DPP1:transmit_data NOP MOV R1, DPP1:transmit_counter; move the transmit_counter into R1 MOVB RL2, DPP2:message_length ; Go through the loop 12 times :: The below add makes the value in R0 point to what ever it was ;; pointing to plus an offset which is in R1 ADD R0, R1; increment the data pointer NOP ;; The problem that I encountered was that I was trying to ;; do a MOV from memory but the data type that was in memory ;; was a BYTE so the computer screwed up. MOVB SOTBUF, [R0] NOP ; Increment the transmit counter register ADDB RL1, #01h MOVB DPP1:transmit_counter, RL1; move the value into the transmit counter CMPB RL1, DPP2:message_length ; comp current count with final count JMP cc_NZ, exit_routine ; if they are equal then stop sending data end_handler: ; necessary to access an Extended SFR EXTR #1 BCLR SOTBIC.6 : for some reason this register is an E-SFR BSET SORIC.6 : asfd BSET T5IE ; asfdasd EXTR #1 ; kjkj BSET XPOIC.6 ; asdfasd ; MOV DPP1:transmit_counter, ZEROS; reset the counter register AND DPP1:message_transmitting, ZEROS; Wait until all queued messages have transm itted to clear CMP ZEROS, DPP1:message_to_transmit; see if any more messages are waiting to tra nsmit JMP cc Z, exit_routine ; jkj : CALL confirm_message ; jkj exit routine: POP DPP2 POP DPP1 ; Pop all data off the stack POP R3

POP R2 POP R1 POP RO RET transmit_buffer_function ENDP do the CAN JAZZ PROC FAR PUSH RO PUSH R5 PUSH R6 PUSH DPP0 NOP MOV DPPO, #PAG direction_of_transmission NOP MOV RL0, DPP0:direction_of_transmission ; See if it is a transmit frame CMPB RLO, #08h JMP cc_Z, transmit_information; jump ; See if it is a remote frame CMPB RLO, #Oh JMP cc_UC, receive_information JMP exit CAN function receive_information: ; This code makes a message object valid MOV R5, #05555h ; Now Message_object 2 is invalid and can be operate MOV MCR_M2, R5 d on ;; Set the message mask MOVB RH5, DPP0:message_id; jkjk MOVB RL5, DPP0:message_id + 1; jadsf MOV R6, #0EF22h NOP MOV [R6], R5 ;; Generate the Message Configuration Register AND R6, ZEROS AND R5, ZEROS MOVB RL5, DPP0:direction_of_transmission MOVB RL6, DPP0:number_of_bytes SHL R6, #04h ADD R5 R6 MOV MCD M2, R5 ;; put data into data register MOV R5, #DPP0:message_data ADD R5, #01h MOV RH6, [R5] ADD R5, #01h MOV RL6, [R5] MOV DATA_M2, R6 ;; Now reactivate the Message Control Object MOV R5, #06599h ; Valid, requested transmission, receive interrupt e nabled MOV MCR M2, R5 ;MOV P2, #05555h ; test pattern JMP exit CAN function transmit_information: ;; Valid Messages get Sent to the CAN BUS ;; First The Message OBJECT Must be setup. :: Message Object 1 is always used right now ;; First make the message invalid MOV R5, #05955h MOV MCR M1, R5



;

END

;; Set the message mask MOVB RH5, DPP0:message_id MOVB RL5, DPP0:message_id + 1 MOV R6, #0EF12h NOP MOV [R6], R5 ;; Generate the Message Configuration Register AND R6, ZEROS AND R5, ZEROS MOVB RL5, DPP0:direction_of_transmission MOVB RL6, DPP0:number_of_bytes SHL R6, #04h ADD R5, R6 MOV P2, R5 ; Test code MOV MCD_M1, R5 ;; put data into data register MOV R5, #DPP0:message_data ADD R5, #01h MOV RH6, [R5] ADD R5, #01h MOV RL6, [R5] MOV DATA_M1, R6 ;; Now reactivate the Message Control Object MOV R5, #06595h MOV MCR_M1, R5 exit_CAN_function: POP DPP0 POP R6 POP R5 POP RO RET do_the_CAN_JAZZ ENDP serial_transmit ENDS



99/04/02 18:55:28

canmod.asm



10	:55:20	cani	mod.asm		
\$SEGME \$EXTEN \$EXTSF \$EXTME \$NOMOE \$STDNA \$SYMBC NAME c	D R M 166 MES(reg167b.def) LS		canin ENDP setall PROC FAR ;; by using a counter ;; objects along the v PUSH R2 PUSH R4 PUSH R5 AND R5, ZEROS		s objs invalid L of the message
	canin ;	define a common register area of 16 registers The function must be declared Global at the beginning of the module	OR R5, #01h AND R2,ZEROS OR R2,#0EF10h AND R4, ZEROS	; Set counter to 1 for first MO ; Set pointer to MO1	
	•	configures specific Message objects	OR R4, #5555h	; Set R4 to make MObs invalid	
	canmocfg:FAR ;	configures specific message objects	nextreg:MOV [R2],R4 ADD R2,#10h CMPI1 R5,#0Fh	; make all message objects invalid	
canfur canin	· .	codesegment that contains reset int pointer	JMPA CC_NZ,nextreg POP R5 POP R4 POP R2 RET	5	
	<pre>PUSH R1 ;; set all of the CAN cor AND C1CSR,ZEROS ; set c MOV R1, #0043h ; OR C1CSR,R1 ; set cor AND C1BTR, ZEROS ; set E MOV R1, #03447h ; OR C1BTR, R1 ; set Bit AND C1GMS, ZEROS ; set C MOV R1, #0FFFFh ; OR C1GML, R1 ; set GMS AND C1LGML, ZEROS ; set C MOV R1, #0FFFFh AND C1LGML, ZEROS ; set C MOV R1, #0F8FFh AND C1LGML, R1 ; AND C1LMLM, ZEROS OR C1LMLM, R1 ; set CMS OR R1, #0001h ; crashes OR R1, #00041h ; crashes OR R1, #00041h ; set CMS </pre>	<pre>control register to zero Set IE and INIT bits htrol register to R1's value Bit timing register to zero set for 125k operation timing register parameters Global Mask short register to zero EOFF is what DAVE initialize Jpper global mask long to zero Jpper global mask upper mask of last register Jower mask of last register Sets all of the CAN registers to off Configures specific Message Objects </pre>	RET setall ENDP canfunc ENDS END		

99/05/06 15:43:38



canmo.asm

SSEGMENTED SEXTEND SEXTSFR SEXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) SSYMBOLS NAME canmo ; declare bank of 16 global registers RBANK1 COMREG R0-R15 GLOBAL canmocfg SECTION CODE can_module ASSUME DPP3:SYSTEM canmocfg PROC FAR PUSH R1 PUSH R2 PUSH R3 ;; Now set specific CAN control Registers ;; initialize message object 1 ;; initializing this object to be invalid does or removing the code until ;; the comment "Setup CAN interrupt and Initialize " does ;; nothing to prevent the occurrance of the interrupt for the CAN system ; start of Message Object 1 MOV R2, #MCR_M1 AND R1, ZEROS ; This MO is inactive and will be controlled from the PC OR R1, #5555h MOV [R2],R1 ; set MO1's Control register ; point to Upper Arbitration register ADD R2,#2h AND R3, ZEROS ; set R3 to ; message id for message object 1 OR R3, #0003h ; message id = #0003hMOV [R2],R3 ; Point to the Lower Arbitration Register ADD R2, #2h ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; put OAAh into first data byte and set to transmit OR R1, #0038h ; Databyte(0) = 0 and Set to receive and 3 bytes of da MOV MCD_M1,R1 ta ; fill the Data of the MO with Zeros MOV DATA_M1, ZEROS ;; set up second message object to be used with receive objects ; start of Message Object 2 MOV R2, #MCR_M2 AND R1, ZEROS ; Generate a Receive Interrupt if this message object ac OR R1, #05555h tivates ; set MO2's Control register MOV [R2],R1 ; point to Upper Arbitration register ADD R2, #2h AND R3, ZEROS ; set R3 to ; message id for message object 2 OR R3, #0003h ; message id = #0003h MOV [R2], R3 ; Point to the Lower Arbitration Register ADD R2, #2h ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; This guy is a receive object OR R1, #0030h ; Databyte(0) = 0 and Set to receive and 3 bytes of da MOV MCD_M2, R1 ta ; fill the Data of the MO with Zeros MOV DATA_M2, ZEROS POP R3 POP R2 POP R1

RET canmocfg ENDP can_module ENDS END

99/04/02 18:51:20	canint.asm		
<pre>\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS</pre>		;; Start building the me MOV R1, R11 AND R1, #0F0h SHL R1, #04h MOVB RL1, #0A0h	; jkasdjfjfkdls essage for serial transmission ; Isolate Data Length Code ; Position it in RH1 ; Move message start bit into place
NAME canint RBANK1 COMREG R0-R15 ; declare bank of 16 global registers		MOV R2, R11	part of the word the Direction of transmission ; Copy into R1 ; Isolate the Direction of the data
ASSUME DPP3:SYSTEM EXTERN message_transmitting:WORD; from serialFebruary EXTERN message_to_transmit:WORD		MOV R3, R13 MOVB RH2, RH3	; Start breaking down the message ID ; Finish Word 2
EXTERN confirm_message:FAR GLOBAL CAN_message_BYTES	S into :	MOVB RH3, #00h	; Start Word 3 ; The First Byte of Data is Always ZERO so Move ZERO
<pre>can_interrupt_data SECTION DATA WORD GLOBAL 'RAM' CAN_message_BYTES LABEL BYTE CAN_message_word_1 DSW 1 CAN_message_word_2 DSW 1 CAN_message_word_3 DSW 1 CAN_message_word_4 DSW 1 CAN_message_word_5 DSW 1 CAN_message_word_6 DSW 1 CAN_message_word_7 DSW 1 CAN_message_word_7 DSW 1</pre>			; Start Word 4 st ZERO therefore don't use a register
CAN_message_word_8 DSW 1 can_interrupt_data ENDS		;; Now compute the Check AND R0, ZEROS AND R9, ZEROS	
can_interrupts SECTION CODE can_receive_interrupt PROC TASK INTNO=040h ORG 0100h		;; Don't user RHI in the MOVB RLO, RH1 ADD R9, R0	e computation of the Checksum ; BYTe to word conversion ; add the Data Length Code to the Checksum
CALL can_interrupt_handler RETI can_receive_interrupt ENDP		AND R0, ZEROS MOVB RL0, RH2 ADD R9, R0	; Reset the byte to word conversion buffer ; add the Direction of transmission to Checksum
can_interrupt_handler PROC FAR PUSH R0 PUSH R1 PUSH R2 PUSH R3 PUSH R3	um	AND R0, ZEROS MOVB RL0, RL2 ADD R9, R0	; add the upper byte of the message id to the checks
PUSH R4 PUSH R5 PUSH R6 PUSH R7 PUSH R8	um	AND R0, ZEROS MOVB RL0, RH3 ADD R9, R0	; add the lower byte of the message id to the checks
PUSH R9 PUSH R10 PUSH R11 PUSH R12 PUSH R12	um	AND RO, ZEROS MOVB RLO, RL3 ADD R9, R0	; add the lower byte of the message id to the checks
PUSH R13 PUSH DPP0 PUSH DPP1 PUSH DPP2 MOV DPP0, #PAG CAN_interrupt_data		AND RO, ZEROS MOVB RLO, RH4 ADD R9, RO	;jk ; add the upper byte of the message data to checksum
MOV R0, #05555h ; deactive code MOV MCR_M2, R0 ; Deactive the Second Message Object AND R7, ZEROS		AND R0, ZEROS MOVB RL0, RL4 ADD R9, R0	; add lower byte of the data to checksum
MOV R11, MCD_M2 ; Moves DLC and DIR into Lower Byte and DATA by upper byte MOV R12, DATA_M2 ; Moves DATA byte 1 into RL2 and DATA b o RH2 MOV R13, MID_M2 ; Moves the Message ID into Register 8		AND R0, ZEROS MOV R6, R9 AND R5, ZEROS MOV RH5, RH6	; Move the checksum into a byte addressable register ; Move the upper byte of the checksum into R5



MOV RL6

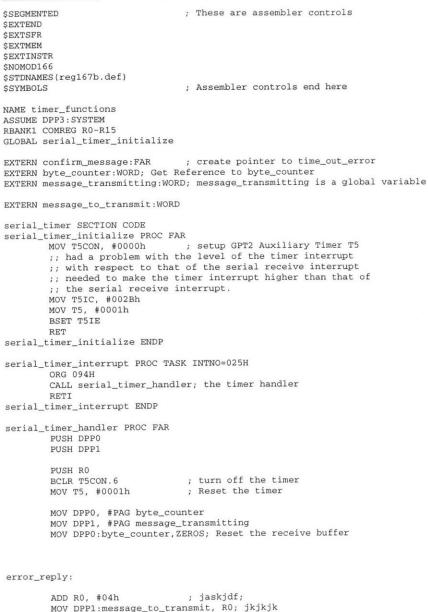
MOV RH6, #OAh

; test



;; THE CHECKSUM IS NOW COMPUTED ;; THE CAN MESSAGE IS NOW COMPLETED IN REGISTERS R1 THROUGH R8 ;; Now put the CAN message into memory MOV DPP0:CAN_message_word_1, R1 ; put data into memory MOV DPP0:CAN_message_word_2, R2 ; put data into memory MOV DPP0:CAN_message_word_3, R3 ; put data into memory MOV DPP0:CAN_message_word_4, R4 ; put data into memory MOV DPP0:CAN_message_word_5, ZEROS ; put data into memory MOV DPP0: CAN_message_word_6, ZEROS ; put data into memory MOV DPP0:CAN_message_word_7, R5 ; put data into memory MOV DPP0:CAN_message_word_8, R6 ; put data into memory MOV R0, #05599h ; Reactive second Message Object MOV MCR_M2, R0 MOV DPP1, #PAG message_transmitting MOV DPP2, #PAG message_to_transmit MOV R0, #08h ADD DPP2:message_to_transmit, R0 CMP ZEROS, DPP1:message_transmitting; test JMP cc_Z, CAN_to_transmit; test JMP cc_UC, exit_can ; test CAN_to_transmit: MOV R0, DPP2:message_to_transmit PUSH RO MOV R1, #08h MOV DPP2:message to transmit, R1 CALL confirm_message ; test POP RO MOV DPP2:message_to_transmit, R0 exit_can: POP DPP2 POP DPP1 POP DPP0 POP R13 POP R12 POP R11 POP R10 POP R9 POP R8 POP R7 POP R6 POP R5 POP R4 POP R3 POP R2 POP R1 POP RO RET can_interrupt_handler ENDP can_interrupts ENDS END

99/05/14 12:09:28



CMP ZEROS, DPP1:message_transmitting JMP cc_NZ, timer_return

CALL confirm_message

timer_return:

timers.asm



POP RO

POP DPP1 POP DPP0 RET serial_timer_handler ENDP serial_timer ENDS END

> MOV R0,#01h MOV DPP1:message_waiting_to_transmit, R0

MOV R0, #02h MOV DPP1:waiting_message, R0



LOCATE main.lno (GENERAL) IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (040000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO main.out

99/05/05 17:48:18

1(71, 1.f

17:48:	18		reg167b.def		
. * * * * * * * * * *	******	*****	MID_M	B DEFA	0EF82h
;** @(#)reg	167b.def	1.10 12/18/97	MID_M		0EF92h
;**			MID_M	A DEFA	0EFA2h
	er definiti	ons for the SAB C167	MID_M	B DEFA	0EFB2h
		s all SFR names and BIT names	MID_M	C DEFA	0EFC2h
;** This fi	le can be	supplied to rm166 and a166 (STDNAMES control)	MID_M	D DEFA	0EFD2h
********	******	*****************	MID_M	e defa	0EFE2h
CICSR		DEFA 0EF00h			
INTID	DEFA	0EF02h			a na basis selatan 1970 na 1 70
C1BTR	DEFA	0EF04h	DP8	DEFR	0FFD6h
CIGMS	DEFA	0EF06h	P8	DEFR	0FFD4h
CIUGML	DEFA	0EF08h	DP7	DEFR	0FFD2h
C1LGML	DEFA	OEFOAh	P7	DEFR	OFFDOh
CIUMLM	DEFA	OEFOCh	DP6	DEFR DEFR	0FFCEh 0FFCCh
CILMLM	DEFA	OEFOEh	P6 DP4	DEFR	OFFCAh
MCR_M1	DEFA	0EF10h	P4	DEFR	0FFC8h
MCR_M2	DEFA	0EF20h	DP3	DEFR	0FFC6h
MCR_M3	DEFA	0EF30h	P3	DEFR	0FFC4h
MCR_M4	DEFA	0EF40h	DP2	DEFR	0FFC2h
MCR_M5	DEFA	0EF50h 0EF60h	P2	DEFR	OFFCOh
MCR_M6	DEFA DEFA	0EF70h	sscco		0FFB2h
MCR_M7 MCR_M8	DEFA	0EF80h	SOCON	DEFR	OFFBOh
MCR_M8 MCR_M9	DEFA	0EF90h	WDTCO		OFFAEh
MCR_M3	DEFA	OEFAOh	TFR	DEFR	OFFACh
MCR_MB	DEFA	OEFBOh	P5	DEFR	0FFA2h
MCR_MC	DEFA	OEFCOh	ADCON	DEFR	OFFAOh
MCR_MD	DEFA	0EFD0h	TIIC	DEFR	0FF9Eh
MCR_ME	DEFA	OEFEOh	TOIC	DEFR	0FF9Ch
MCR_MF	DEFA	OEFFOh	ADEIC	DEFR	0FF9Ah
MCD_M1	DEFA	0EF16h	ADCIC	DEFR	0FF98h
MCD_M2	DEFA	0EF26h	CC151		0FF96h
MCD_M3	DEFA	0EF36h	CC14I		0FF94h
MCD_M4	DEFA	0EF46h	CC13I		0FF92h
MCD_M5	DEFA	0EF56h	CC12I		0FF90h
MCD_M6	DEFA	0EF66h	CC11I		OFF8Eh
MCD_M7	DEFA	0EF76h	CC10I		0FF8Ch
MCD_M8	DEFA	0EF86h	CC9IC	DEFR	0FF8Ah 0FF88h
MCD_M9	DEFA	0EF96h	CC8IC	DEFR DEFR	0FF86h
MCD_MA	DEFA	0EFA6h	CC7IC CC6IC	DEFR	0FF84h
MCD_MB	DEFA	0EFB6h	CC5IC		0FF82h
MCD_MC	DEFA	0EFC6h	CC4IC	DEFR	0FF80h
MCD_MD	DEFA	0EFD6h	CC3IC		0FF7Eh
MCD_ME	DEFA DEFA	0EFE6h 0EF18h	CC2IC		0FF7Ch
DATA_M1	DEFA	0EF18h 0EF28h	CC1IC		0FF7Ah
DATA_M2 DATA_M3	DEFA	0EF38h	CCOIC		0FF78h
DATA_M3 DATA_M4	DEFA	0EF48h	SSCEI		0FF76h
DATA_M4 DATA_M5	DEFA	0EF58h	SSCRI		0FF74h
DATA_M6	DEFA	0EF68h	SSCTI		0FF72h
DATA_M7	DEFA	0EF78h	SOEIC		0FF70h
DATA_M8	DEFA	0EF88h	SORIC		0FF6Eh
DATA_M9	DEFA	0EF98h	SOTIC		0FF6Ch
DATA_MA	DEFA	0EFA8h	CRIC	DEFR	0FF6Ah
DATA_MB	DEFA	0EFB8h	TGIC	DEFR	0FF68h
DATA_MC	DEFA	0EFC8h	T5IC	DEFR	0FF66h
DATA_MD	DEFA	0EFD8h	T4IC	DEFR	0FF64h
DATA_ME	DEFA	0EFE8h	T3IC	DEFR	0FF62h
MID_M1	DEFA	0EF12h	T2IC	DEFR	0FF60h
MID_M2	DEFA	0EF22h	CCM3	DEFR	0FF58h
MID_M3	DEFA	0EF32h	CCM2	DEFR	OFF56h OFF54b
MID_M4	DEFA	0EF42h	CCM1	DEFR	0FF54h 0FF52h
MID_M5	DEFA	0EF52h	CCM0	DEFR DEFR	0FF52h 0FF50h
MID_M6	DEFA	0EF62h	T01CC T6CON		0FF48h
MID_M7	DEFA	0EF72h	16001	DEFR	0114011

99/05/05 17:48:18

T5CON

T4CON

T3CON

T2CON

CCM7

CCM6

CCM5

CCM4

P1H

P1L

POH

POL

PECC7

PECC6

PECC5

PECC4

PECC3

PECC2

PECC1

PECC0

SRCP0

DSTP0

SRCP1

DSTP1

SRCP2

DSTP2

SRCP3

DSTP3

SRCP4

DSTP4

SRCP5

DSTP5

SRCP6

DSTP6

SRCP7

DSTP7

SOBG

WDT

ADDAT

CC15

CC14

CC13

CC12

CC11

CC10

CC9

CC8

CC7

CC6

CC5

CC4

CC3

CC2

CC1

CC0

CC31

CC30

CC29

CC28

CC27

SORBUF

SOTBUF

T78CON

PWMCON1

PWMCON0

0FF46h

0FF44h

0FF42h

0FF40h

0FF32h

0FF30h

0FF28h

0FF26h

0FF24h

0FF22h

0FF20h

0FF06h

0FF04h

0FF02h

0FF00h

OFECEh

0FECCh

OFECAh

0FEC8h

0FEC6h

0FEC4h

0FEC2h

OFECOh

0FCE0h

0FCE2h

0FCE4h

0FCE6h

0FCE8h

OFCEAh

0FCECh

OFCEEh

OFCFOh

0FCF2h

0FCF4h

0FCF6h

0FCF8h

OFCFAh

OFCFCh

OFCFEh

0FEB4h

OFEAOh

0FE9Eh

0FE9Ch

0FE9Ah

0FE98h

0FE96h

0FE94h

0FE92h

0FE90h

0FE8Eh

0FE8Ch

0FE8Ah

0FE88h

0FE86h

0FE84h

0FE82h

0FE80h

0FE7Eh

0FE7Ch

0FE7Ah

0FE78h

0FE76h

OFEB2h, r

OFEBOh, w

OFEAEh, r

DEFR

DEFA

DEFA DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFA

DEFR

OFE74h

0FE72h

0FE70h

0FE6Eh

0FE6Ch

0FE6Ah

0FE68h

0FE66h

0FE64h

0FE62h

0FE60h

0FE56h

0FE54h

0FE52h

0FE50h

0FE4Ah

0FE48h

0FE46h

0FE44h

0FE42h

0FE40h

0FE36h

0FE34h

0FE32h

0FE30h

0F1D6h

0F1D2h

OF1CEh

0F1C6h

0F1C4h

0F1C2h

0F1C0h

0F19Ch

OF19Eh

0F196h

0F18Eh

0F186h

OF17Eh

0F17Ch

0F17Ah

0F194h

0F18Ch

0F184h

0F178h

0F176h

0F174h

0F172h

0F170h

0F16Eh

0F16Ch

0F16Ah

0F168h

0F166h

0F164h

0F162h

0F160h

0F108h

0F106h

0F104h

0F102h

0F100h

OFOB4h

		1.1		
	197	1		
	1	di Q		
		anii		

10010/0	1001	
	0026	DEED
	CC26	DEFR
	CC25	DEFR
	CC24	DEFR
	CC23	DEFR
	CC22	DEFR
	CC21	DEFR
	CC20	DEFR
	CC19	DEFR
	CC18	DEFR
	CC17	DEFR
	CC16	DEFR
	TIREL	DEFR
	TOREL	DEFR
	T1	DEFR
	то	DEFR
	CAPREL	DEFR
	Т6	DEFR
	Т5	DEFR
	Т4	DEFR
	Т3	DEFR
	T2	DEFR
	PW3	DEFR
		DEFR
	PW2	DEFR
	PW1	
	PW0	DEFR
	; Extended sfr	: area
	ODP8	DEFR
	ODP7	DEFR
	ODP6	DEFR
	ODP8 ODP3	DEFR
		DEFR
	PICON	
	ODP2	DEFR
	EXICON	DEFR
	SOTBIC	DEFR
	XP3IC	DEFR
	XP2IC	DEFR
	XP1IC	DEFR
	XPOIC	DEFR
	PWMIC	DEFR
	T8IC	DEFR
	T7IC	DEFR
	CC31IC	DEFR
	CC30IC	DEFR
	CC29IC	DEFR
	CC28IC	DEFR
	CC27IC	DEFR
	CC26IC	DEFR
	CC25IC	DEFR
	CC24IC	DEFR
	CC23IC	DEFR
	CC22IC	DEFR
	CC21IC	DEFR
	CC20IC	DEFR
	CC19IC	DEFR
	CC18IC	DEFR
	CC17IC	DEFR
	CC16IC	DEFR
	RPOH	DEFR
	DP1H	DEFR
		DEFR
	DP1L	
	DPOH	DEFR
	DPOL	DEFR
	SSCBR	DEFR

17:48:18		
SSCRB	DEFR	0F0B2h
SSCTB	DEFR	OFOBOh
ADDAT2	DEFR	OFOAOh
T8REL	DEFR	0F056h
T7REL	DEFR	0F054h
Т8	DEFR	0F052h
т7	DEFR	0F050h
PP3	DEFR	0F03Eh
PP2	DEFR	0F03Ch
PP1	DEFR	0F03Ah
PPO	DEFR	0F038h
PT3	DEFR	0F036h
PT2	DEFR	0F034h
PT1	DEFR	0F032h
PTO	DEFR	0F030h
; Bit names		
CC0IO	DEFB	P2.0
CC1IO	DEFB	P2.1
CC210	DEFB	P2.2
CC310	DEFB	P2.3
CC410	DEFB	P2.4
CC510	DEFB	P2.5
CC6I0	DEFB	P2.6
CC710	DEFB	P2.7
CC810	DEFB	P2.8
CC910	DEFB	P2.9
CC10IO	DEFB	P2.10
CC11IO	DEFB	P2.11
CC12I0	DEFB	P2.12
CC13I0	DEFB	P2.13
CC14I0	DEFB	P2.14
CC15I0	DEFB	P2.15
EXOIN	LIT	'CC0I0'
EX1IN	LIT	'CC1IO'
EX2IN	LIT	'CC2I0'
EX3IN	LIT	'CC3I0'
TOIN	DEFB	P3.0
TGOUT	DEFB	P3.1
CAPIN	DEFB	P3.2
TJOUT	DEFB	P3.3
T3EUD	DEFB	P3.4
T2IN	DEFB	P3.7
TJIN	DEFB	P3.6
T4IN	DEFB	P3.5
SSDI	DEFB	P3.8
SSDO	DEFB	P3.9
TXD0	DEFB	P3.10
RXD0	DEFB	P3.11
SSCLK	DEFB	P3.13
CLKOUT	DEFB	P3.15
A16	DEFB	P4.0 P4.1
A17	DEFB	P4.1 P4.2
A18	DEFB	P4.2 P4.3
A19	DEFB	P4.3 P4.4
A20	DEFB	P4.4 P4.5
A21	DEFB	P4.5 P4.6
A22 A23	DEFB DEFB	P4.0 P4.7
	DEFB	P5.0
ANO	DEFB	P5.0 P5.1
AN1	DEFB	P5.1 P5.2
AN2	DEFB	EJ.2

99/05/05

reg167b.def	reg1	67b.o	lef
-------------	------	-------	-----

def		
AN3	DEFB	P5.3
AN4	DEFB	P5.4
AN5	DEFB	P5.5
AN6	DEFB	P5.6
AN7	DEFB	P5.7 P5.8
AN8	DEFB DEFB	P5.8 P5.9
AN9 AN10	DEFB	P5.10
AN11	DEFB	P5.11
AN12	DEFB	P5.12
AN13	DEFB	P5.13
AN14	DEFB	P5.14
AN15	DEFB	P5.15
T6EUD	LIT	'AN10' 'AN11'
T5EUD	LIT LIT	'AN12'
T6IN T5IN	LIT	'AN13'
T4EUD	LIT	'AN14'
T2EUD	LIT	'AN15'
POUT0	DEFB	P7.0
POUT1 POUT2	DEFB DEFB	P7.1 P7.2
POUT3	DEFB	P7.3
CC28I0	DEFB	P7.4
CC2910	DEFB	P7.5
CC30I0	DEFB	P7.6
CC31I0	DEFB	P7.7
CC16I0	DEFB	P8.0
CC17I0	DEFB	P8.1
CC18I0	DEFB	P8.2
CC19I0	DEFB	P8.3
CC20I0	DEFB	P8.4 P8.5
CC21I0 CC22I0	DEFB DEFB	P8.6
CC2310	DEFB	P8.7
000010		
TOM	DEFB	T01CON.3
TOR	DEFB	T01CON.6
T1M	DEFB	T01CON.11
TIR	DEFB	T01CON.14
T7M	DEFB	T78CON.3 T78CON.6
T7R T8M	DEFB DEFB	T78CON.8
T8R	DEFB	T78CON.14
ACC0	DEFB	CCM0.3
ACC1	DEFB	CCM0.7
ACC2	DEFB	CCM0.11
ACC3	DEFB	CCM0.15
ACC4	DEFB	CCM1.3
ACC5	DEFB	CCM1.7 CCM1.11
ACC6 ACC7	DEFB DEFB	CCM1.11 CCM1.15
ACC8 ACC9	DEFB DEFB	CCM2.3 CCM2.7
ACC10	DEFB	CCM2.11
ACC11	DEFB	CCM2.15
ACC12	DEFB	CCM3.3
ACC13	DEFB	CCM3.7

99/05/05 17:48:18		
ACC14	DEFB	CCM3.11
ACC15	DEFB	CCM3.15
ACC16	DEFB	CCM4.3
ACC17	DEFB	CCM4.7
ACC18	DEFB	CCM4.11
ACC19	DEFB	CCM4.15
ACC20	DEFB	CCM5.3
ACC21	DEFB	CCM5.7
ACC22	DEFB	CCM5.11
ACC23	DEFB	CCM5.15
ACC24	DEFB	CCM6.3
ACC25	DEFB	CCM6.7
ACC26	DEFB	CCM6.11
ACC27	DEFB	CCM6.15
ACC28	DEFB	CCM7.3
ACC29	DEFB	CCM7.7
ACC30	DEFB	CCM7.11
ACC31	DEFB	CCM7.15
T2R	DEFB	T2CON.6
T2UD	DEFB	T2CON.7
T2UDE	DEFB	T2CON.8
T3R	DEFB	T3CON.6
T3UD	DEFB	T3CON.7
T3UDE	DEFB	T3CON.8
T3OE	DEFB	T3CON.9
T3OTL	DEFB	T3CON.10
T4R	DEFB	T4CON.6
T4UD	DEFB	T4CON.7
T4UDE	DEFB	T4CON.8
T5R	DEFB	T5CON.6
T5UD	DEFB	T5CON.7
T5UDE	DEFB	T5CON.8
T5CLR	DEFB	T5CON.14
T5SC	DEFB	T5CON.15
T6R	DEFB	T6CON.6
T6UD	DEFB	T6CON.7
T6UDE	DEFB	T6CON.8
T6OE	DEFB	T6CON.9
T6OTL	DEFB	T6CON.10
T6SR	DEFB	T6CON.15
T2IE T2IR T3IE T3IR T4IE T4IR T5IE T5IR	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T2IC.6 T2IC.7 T3IC.6 T3IC.7 T4IC.6 T4IC.7 T5IC.6 T5IC.7
T6IE	DEFB	T6IC.6
T6IR	DEFB	T6IC.7
CRIE	DEFB	CRIC.6
CRIR	DEFB	CRIC.7
SOTIE	DEFB	SOTIC.6

Teg107	0.del		
	SOTIR	DEFB	SOTIC.7
	SORIE	DEFB	SORIC.6
	SORIR	DEFB	SORIC.7
	SOEIE	DEFB	SOEIC.6
	SOEIR	DEFB	SOEIC.7
	SOTBIE	DEFB	SOTBIC.6
	SOTBIE	DEFB	SOTBIC.0
	SUIDIK	DEFB	SUIDIC. /
	SSCTIE	DEFB	SSCTIC.6
	SSCTIR	DEFB	SSCTIC.7
	SSCRIE	DEFB	SSCRIC.6
	SSCRIR	DEFB	SSCRIC.7
	SSCEIE	DEFB	SSCEIC.6
	SSCEIR	DEFB	SSCEIC.7
	SSCTE	LIT	'SSCTEN'
	SSCRE	LIT	'SSCREN'
	SSCPE	LIT	'SSCPEN'
	SSCBE	LIT	'SSCBEN'
	22075		00010
	CCOIE	DEFB	CCOIC.6
	CCOIR	DEFB	CCOIC.7 CCIIC.6
	CC1IE	DEFB	
	CC1IR	DEFB	CC1IC.7 CC2IC.6
	CC2IE	DEFB DEFB	CC2IC.8
	CC2IR CC3IE	DEFB	CC3IC.6
	CC3IR	DEFB	CC3IC.7
	CC4IE	DEFB	CC4IC.6
	CC4IE CC4IR	DEFB	CC4IC.7
	CC5IE	DEFB	CC5IC.6
	CC5IR	DEFB	CC5IC.7
	CC6IE	DEFB	CC6IC.6
	CC6IR	DEFB	CC6IC.7
	CC7IE	DEFB	CC7IC.6
	CC7IR	DEFB	CC7IC.7
	CC8IE	DEFB	CC8IC.6
	CC8IR	DEFB	CC8IC.7
	CC9IE	DEFB	CC9IC.6
	CC9IR	DEFB	CC9IC.7
	CC10IE	DEFB	CC10IC.6
	CC10IR	DEFB	CC10IC.7
	CC11IE	DEFB	CC11IC.6
	CC11IR	DEFB	CC11IC.7
	CC12IE	DEFB	CC12IC.6
	CC12IR	DEFB	CC12IC.7
	CC13IE	DEFB	CC13IC.6
	CC13IR	DEFB	CC13IC.7 CC14IC.6
	CC14IE	DEFB	CC141C.8
	CC14IR CC15IE	DEFB DEFB	CC14IC.7
	CC15IE CC15IR	DEFB	CC15IC.7
	CC16IE	DEFB	CC16IC.6
	CC16IR	DEFB	CC16IC.7
	CC17IE	DEFB	CC17IC.6
	CC17IE CC17IR	DEFB	CC17IC.7
	CC18IE	DEFB	CC18IC.6
	CC18IR	DEFB	CC18IC.7
	CC19IE	DEFB	CC19IC.6
	CC19IR	DEFB	CC19IC.7
	CC20IE	DEFB	CC20IC.6
	CC20IR	DEFB	CC20IC.7
	CC21IE	DEFB	CC21IC.6
	CC21IR	DEFB	CC21IC.7
	CC22IE	DEFB	CC22IC.6

	and the second	
	- 197 	

reg167b.def

CC22IRDEFBCC22IC.7CC23IEDEFBCC23IC.6CC23IRDEFBCC23IC.6CC24IEDEFBCC24IC.6CC24IRDEFBCC24IC.7CC25IEDEFBCC25IC.6CC25IRDEFBCC26IC.7CC26IEDEFBCC26IC.7CC27IEDEFBCC26IC.7CC27IEDEFBCC26IC.7CC27IEDEFBCC28IC.7CC28IEDEFBCC28IC.7CC29IEDEFBCC201C.7CC30IEDEFBCC30IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC31IC.7ADCIEDEFBADCIC.6ADCIRDEFBADCIC.7ADEIEDEFBADCIC.7ADEIEDEFBTOIC.6TOIRDEFBTOIC.7T1IEDEFBTOIC.7T1IEDEFBTOIC.6T0IRDEFBTSIC.6T7IRDEFBTSIC.6T8IRDEFBTSIC.6T8IRDEFBTSIC.6T8IRDEFBADCON.7ADSTDEFBADCON.7ADSYDEFBADCON.10ADCINDEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.13STKOFDEFBTFR.13SORPDEFBSOCON.14SOPEDEFBSOCON.10SOOPEDEFBSOCON.15	99/05/05 17:48:18		
CC23IEDEFBCC23IC.6CC23IRDEFBCC23IC.7CC24IEDEFBCC24IC.7CC25IEDEFBCC25IC.6CC25IRDEFBCC25IC.7CC26IEDEFBCC26IC.7CC27IEDEFBCC27IC.6CC27IRDEFBCC27IC.7CC28IEDEFBCC28IC.7CC29IRDEFBCC29IC.6CC29IRDEFBCC29IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC30IC.7CC31IRDEFBCC30IC.7CC31IRDEFBCC31IC.7CC31IRDEFBADCIC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIEDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7T1IEDEFBTOIC.7T1IEDEFBTOIC.7T3IEDEFBTOIC.7T3IEDEFBTOIC.7ADSTDEFBTSIC.7ADSTDEFBADCON.7ADBSYDEFBADCON.7ADCINDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.7STKUFDEFBTFR.7STKUFDEFBTFR.13SORENDEFBSOCON.6SOOENDEFBSOCON.7SOPEDEFBSOCON.7 <tr< td=""><td>000070</td><td>DEED</td><td>CC22TC 7</td></tr<>	000070	DEED	CC22TC 7
CC23IRDEFBCC23IC.7CC24IEDEFBCC24IC.6CC24IRDEFBCC24IC.7CC25IEDEFBCC25IC.7CC26IEDEFBCC26IC.6CC26IRDEFBCC26IC.7CC27IEDEFBCC27IC.7CC28IEDEFBCC27IC.7CC28IEDEFBCC28IC.6CC29IRDEFBCC29IC.7CC30IEDEFBCC29IC.7CC30IEDEFBCC30IC.7CC30IEDEFBCC30IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC31IC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIEDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7T1IEDEFBTTC.7T7IEDEFBTTC.7T8IEDEFBTTC.7ADSTDEFBADCON.7ADSTDEFBADCON.7ADSTDEFBADCON.8ADWRDEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.1SOSTPDEFBSOCON.4SOPENDEFBSOCON.10SOOENDEFBSOCON.7SOPENDEFBSOCON.11SOSTPDEFBSOCON.12SOERSDEFBSOCON.14SOPEDEFBSOCON.14SOPEDEFBSOCON.14 <td></td> <td></td> <td></td>			
CC24IEDEFBCC24IC.6CC24IRDEFBCC24IC.7CC25IEDEFBCC25IC.7CC25IRDEFBCC26IC.6CC25IRDEFBCC26IC.7CC27IEDEFBCC26IC.7CC27IEDEFBCC28IC.6CC27IRDEFBCC28IC.7CC29IEDEFBCC28IC.7CC29IEDEFBCC29IC.7CC30IEDEFBCC30IC.6CC30IRDEFBCC30IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC31IC.7ADCIEDEFBADCIC.6ADCIRDEFBADCIC.7ADEIEDEFBADCIC.7ADEIEDEFBADCIC.7ADEIEDEFBTOIC.7T1IEDEFBTOIC.7T1IEDEFBTOIC.7T3IEDEFBTTC.6T7IRDEFBTTC.7T8IEDEFBTSC.7ADSTDEFBADCON.7ADSYDEFBADCON.8ADWRDEFBADCON.8ADWRDEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.1STKOFDEFBTFR.13STKOFDEFBTFR.13SOSTPDEFBSOCON.10SOOPEDEFBSOCON.11SOSTPDEFBSOCON.12SOPEDEFBSOCON.12SOPEDEFBSOCON.12SODDDEFBSOCON.13SOLBDEFBSOCON.14 <tr< td=""><td></td><td></td><td></td></tr<>			
CC24IRDEFBCC24IC.7CC25IEDEFBCC25IC.6CC25IRDEFBCC25IC.7CC26IEDEFBCC26IC.7CC26IRDEFBCC26IC.7CC27IEDEFBCC26IC.7CC27IEDEFBCC26IC.7CC28IEDEFBCC28IC.7CC29IEDEFBCC29IC.6CC29IRDEFBCC29IC.7CC30IEDEFBCC30IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC31IC.6CC31IRDEFBCC31IC.7ADCIEDEFBADCIC.6ADCIRDEFBADCIC.7ADEIEDEFBTOIC.7ADEIEDEFBTOIC.7T1IEDEFBTOIC.7T1IEDEFBTOIC.7T1IEDEFBTTC.7T8IEDEFBTSIC.7ADSTDEFBADCON.7ADBSYDEFBADCON.7ADCRQDEFBADCON.9ADCINDEFBTFR.1ADCRQDEFBTFR.1ILLOPADEFBTFR.1SOSTPDEFBTFR.13STKOFDEFBTFR.13SORENDEFBSOCON.4SOPENDEFBSOCON.6SOOENDEFBSOCON.10SOODDDEFBSOCON.12SOBRSDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14	CC23IR	DEFB	
CC25IEDEFBCC25IC.6CC25IRDEFBCC25IC.7CC26IEDEFBCC26IC.7CC26IRDEFBCC26IC.7CC27IEDEFBCC27IC.6CC27IRDEFBCC28IC.7CC28IEDEFBCC29IC.6CC29IRDEFBCC29IC.7CC30IEDEFBCC29IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC30IC.7CC31IEDEFBCC31IC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIEDEFBTOIC.6ADEIRDEFBTOIC.7TIEDEFBTOIC.7TIEDEFBTOIC.7TIEDEFBTOIC.7TIEDEFBTOIC.7TRDEFBTTIC.7TRDEFBTTIC.7TRDEFBTTIC.7TRDEFBTSIC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.1SOSTPDEFBSOCON.3SORENDEFBSOCON.11SOSTPDEFBSOCON.12SOOPEDEFBSOCON.12SOOPEDEFBSOCON.12SOCONDEFBSOCON.14SOCODDEFBSOCON.14SOCDDEFBSOCON.14SOLE	CC24IE	DEFB	CC24IC.6
CC25IRDEFBCC25IC.7CC26IEDEFBCC26IC.6CC26IRDEFBCC26IC.7CC27IEDEFBCC27IC.7CC28IEDEFBCC28IC.7CC28IRDEFBCC29IC.6CC29IRDEFBCC29IC.7CC30IEDEFBCC29IC.7CC30IEDEFBCC29IC.7CC30IEDEFBCC30IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC31IC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIRDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TIEDEFBTOIC.7TIEDEFBTTIC.6TIRDEFBTTIC.7T7IEDEFBTSIC.6T8IRDEFBTSIC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.8ADWRDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.13STKOFDEFBTFR.13STKOFDEFBTFR.13SOSTPDEFBSOCON.10MDTINDEFBSOCON.11SOSTPDEFBSOCON.12SOPEDEFBSOCON.7SOPEDEFBSOCON.7SOPEDEFBSOCON.12SOOEDEFBSOCON.14SOP	CC24IR	DEFB	CC24IC.7
CC25IRDEFBCC25IC.7CC26IEDEFBCC26IC.6CC26IRDEFBCC26IC.7CC27IEDEFBCC27IC.7CC28IEDEFBCC28IC.7CC28IRDEFBCC29IC.6CC29IRDEFBCC29IC.7CC30IEDEFBCC29IC.7CC30IEDEFBCC29IC.7CC30IEDEFBCC30IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC31IC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIRDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TIEDEFBTOIC.7TIEDEFBTTIC.6TIRDEFBTTIC.7T7IEDEFBTSIC.6T8IRDEFBTSIC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.8ADWRDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.13STKOFDEFBTFR.13STKOFDEFBTFR.13SOSTPDEFBSOCON.10MDTINDEFBSOCON.11SOSTPDEFBSOCON.12SOPEDEFBSOCON.7SOPEDEFBSOCON.7SOPEDEFBSOCON.12SOOEDEFBSOCON.14SOP		DEFB	CC25IC.6
CC2 6TEDEFBCC2 6TC. 6CC2 6TRDEFBCC2 6TC. 7CC2 7TEDEFBCC2 7TC. 6CC2 7TRDEFBCC2 7TC. 7CC2 8TEDEFBCC2 8TC. 7CC2 9TEDEFBCC2 9TC. 7CC3 0TEDEFBCC3 0TC. 7CC3 0TEDEFBCC3 0TC. 7CC3 0TEDEFBCC3 0TC. 7CC3 0TEDEFBCC3 0TC. 7CC3 1TEDEFBCC3 0TC. 7CC3 1TEDEFBCC3 0TC. 7CC3 1TEDEFBCC3 1TC. 7ADC 1EDEFBADCIC. 6ADC 1RDEFBADEIC. 7ADE 1EDEFBADEIC. 7ADE 1EDEFBTOIC. 7T0 1EDEFBTOIC. 7T1 1EDEFBTOIC. 7T7 1EDEFBTTC. 7T8 1EDEFBTTC. 7ADSTDEFBADCON. 7ADSTDEFBADCON. 8ADWRDEFBADCON. 9ADC 1NDEFBTFR. 1ILLOPADEFBTFR. 1ILLOPADEFBTFR. 1ILLOPADEFBTFR. 1STKOFDEFBTFR. 13STKOFDEFBTFR. 13SOSTPDEFBSOCON. 1SOSTPDEFBSOCON. 1SOPENDEFBSOCON. 1SOOPENDEFBSOCON. 1SOOPEDEFBSOCON. 1SOCOPDEFBSOCON. 1SOCOPDEFBSOCON. 1SOCOPDEFBS			
CC2 6TRDEFBCC2 6TC.7CC2 7TEDEFBCC2 7TC.6CC2 7TRDEFBCC2 7TC.7CC2 8TEDEFBCC2 8TC.6CC2 9TEDEFBCC2 8TC.7CC2 9TEDEFBCC2 9TC.6CC2 9TRDEFBCC3 0TC.7CC3 0TEDEFBCC3 0TC.7CC3 0TEDEFBCC3 0TC.6CC3 1TEDEFBCC3 1TC.6CC3 1TRDEFBCC3 1TC.7ADCTEDEFBADCTC.6ADCTRDEFBADCTC.7ADETEDEFBADCTC.7ADETEDEFBTOTC.7TOTEDEFBTOTC.7TTTEDEFBTTC.7TTTEDEFBTTC.7TSTEDEFBTTC.7TSTEDEFBTSTC.7ADSTDEFBADCON.7ADDSYDEFBADCON.8ADWRDEFBADCON.9ADCTNDEFBADCON.10ADCRQDEFBTFR.0TLLUOPADEFBTFR.1TLLOPADEFBTFR.1STKUFDEFBTFR.13STKOFDEFBTFR.13SORENDEFBSOCON.4SOPENDEFBSOCON.13SORENDEFBSOCON.14SOPEDEFBSOCON.12SOBSDEFBSOCON.14SOLBDEFBSOCON.14SOLBDEFBSOCON.14			
CC27TEDEFBCC27IC.6CC27IRDEFBCC27IC.7CC28TEDEFBCC28IC.7CC28TEDEFBCC28IC.7CC29TEDEFBCC29IC.6CC29TEDEFBCC29IC.7CC30TEDEFBCC30IC.6CC30TRDEFBCC31IC.6CC31TRDEFBCC31IC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADETEDEFBADCIC.7ADETEDEFBTOTC.7TOTEDEFBTOTC.7TOTEDEFBTTC.7TTIEDEFBTTC.7TTIEDEFBTTC.7TREDEFBTTC.7TREDEFBTTC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBTFR.0TLLINADEFBTFR.1ILLOPADEFBTFR.1UNDOPCDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.13SOSTPDEFBSOCON.11SOSTPDEFBSOCON.12SOPENDEFBSOCON.13SORENDEFBSOCON.6SOOENDEFBSOCON.7SOPEDEFBSOCON.12SODDDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14			
CC27TRDEFBCC27IC.7CC28IEDEFBCC28IC.6CC28TRDEFBCC28IC.7CC29IEDEFBCC29IC.7CC20IEDEFBCC29IC.7CC30IEDEFBCC29IC.7CC30IEDEFBCC29IC.7CC30IEDEFBCC30IC.7CC31IEDEFBCC31IC.7CC31IEDEFBCC31IC.7ADCIEDEFBADCIC.6ADCIRDEFBADEIC.7ADEIEDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TILEDEFBTOIC.7TIRDEFBTTC.7TRDEFBTTC.6T7IRDEFBTSIC.6T8IRDEFBTSIC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.13SOSTPDEFBSOCON.11SOSTPDEFBSOCON.12SOPENDEFBSOCON.7SOPENDEFBSOCON.7SOPENDEFBSOCON.7SOPEDEFBSOCON.12SOOEDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14 <td></td> <td></td> <td></td>			
CC28IEDEFBCC28IC.6CC28IRDEFBCC28IC.7CC29IEDEFBCC29IC.7CC30IEDEFBCC30IC.6CC30IRDEFBCC30IC.7CC31IEDEFBCC30IC.7CC31IEDEFBCC31IC.7ADCIEDEFBADCIC.6ADCIRDEFBADCIC.7ADEIEDEFBADEIC.7ADEIEDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TILEDEFBTOIC.7TIRDEFBTTC.6TIRDEFBTTC.7T8IEDEFBTTC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.10SOSTPDEFBSOCON.1SOSTPDEFBSOCON.1SOFPDEFBSOCON.7SOPENDEFBSOCON.7SOPEDEFBSOCON.12SOOPEDEFBSOCON.12SODDDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14			
CC28IRDEFBCC28IC.7CC29IEDEFBCC29IC.6CC29IRDEFBCC29IC.7CC30IEDEFBCC30IC.7CC30IRDEFBCC30IC.7CC31IEDEFBCC31IC.6CC31IRDEFBCC31IC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIEDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TILEDEFBTOIC.7TILEDEFBTOIC.7TRDEFBTTIC.6T7IRDEFBTTIC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.10ADCRQDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1STKOFDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.15WDTINDEFBSOCON.4SOSTPDEFBSOCON.6SOCENDEFBSOCON.6SOCENDEFBSOCON.7SORENDEFBSOCON.7SORENDEFBSOCON.7SORENDEFBSOCON.7SOPEDEFBSOCON.7SOCENDEFBSOCON.7SOCENDEFBSOCON.10SOODDDEFBSOCON.14SOLBDEFBSOCON.14			
CC29IEDEFBCC29IC.6CC29IRDEFBCC30IC.7CC30IEDEFBCC30IC.7CC30IEDEFBCC31IC.6CC31IRDEFBCC31IC.6ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIEDEFBADCIC.7ADEIEDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TIEDEFBTTC.7TRDEFBTTC.7TRDEFBTTC.7T8IEDEFBT8IC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.3UNDOPCDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.10SOSTPDEFBSOCON.11SOSTPDEFBSOCON.6SOOENDEFBSOCON.6SOOENDEFBSOCON.7SOPEDEFBSOCON.7SORENDEFBSOCON.7SOPEDEFBSOCON.10SOODDDEFBSOCON.12SOBRSDEFBSOCON.14SOLBDEFBSOCON.14SOLBDEFBSOCON.14			
CC29IRDEFBCC29IC.7CC30IEDEFBCC30IC.6CC30IRDEFBCC30IC.6CC31IEDEFBCC31IC.6CC31IRDEFBCC31IC.7CC31IRDEFBADCIC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIEDEFBADEIC.7ADEIEDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TIEDEFBTIC.7T7IEDEFBTTC.7T8IEDEFBTTC.7T8IEDEFBT8IC.6T8IRDEFBADCON.7ADSTDEFBADCON.7ADSTDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1UNDOPCDEFBTFR.13STKOFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.10WDTINDEFBSOCON.11SOSTPDEFBSOCON.12SOFPDEFBSOCON.7SOPENDEFBSOCON.7SOPEDEFBSOCON.7SOPEDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14			
CC30IEDEFBCC30IC.6CC30IRDEFBCC30IC.7CC31IEDEFBCC30IC.7CC31IEDEFBCC31IC.6CC31IRDEFBCC31IC.7ADCIEDEFBADCIC.7ADEIEDEFBADEIC.6ADEIRDEFBTOIC.6TOIRDEFBTOIC.7T1IEDEFBTOIC.7T1IEDEFBTTC.6T7IRDEFBTTC.7T8IEDEFBTTC.6T8IRDEFBTTC.6ADSTDEFBT8IC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1ILLOPADEFBTFR.1STKOFDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.1SOSTPDEFBSOCON.1SOSTPDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOPEDEFBSOCON.7SOPEDEFBSOCON.7SOCEDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14	CC29IE	DEFB	
CC30IRDEFBCC30IC.7CC31IEDEFBCC31IC.6CC31IRDEFBCC31IC.7ADCIEDEFBADCIC.7ADCIEDEFBADCIC.7ADEIEDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TIEDEFBTOIC.7TIEDEFBTTIC.7TRDEFBTTIC.7TRDEFBTTIC.7TRDEFBTTIC.6T7IRDEFBTTC.6T7IRDEFBTSIC.6T8IRDEFBTSIC.6T8IRDEFBADCON.7ADSTDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBTFR.0ILLINADEFBTFR.2PRTFLTDEFBTFR.2PRTFLTDEFBTFR.13UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBSOCON.4SOSTPDEFBSOCON.6SOCENDEFBSOCON.6SOFEDEFBSOCON.7SOPEDEFBSOCON.10SOODDDEFBSOCON.14SOLBDEFBSOCON.14SOLBDEFBSOCON.14	CC29IR	DEFB	
CC31IEDEFBCC31IC.6CC31IRDEFBCC31IC.7ADCIEDEFBADCIC.6ADCIRDEFBADCIC.7ADEIEDEFBADEIC.6ADEIRDEFBTOIC.7TOIEDEFBTOIC.7TIEDEFBTOIC.7TIEDEFBTTC.6T7IRDEFBTTC.7T8IEDEFBTTC.7ADSTDEFBADCON.7ADBSYDEFBADCON.7ADCINDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.13SOSTPDEFBSOCON.4SOFENDEFBSOCON.6SOCENDEFBSOCON.6SOCENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.6SOCENDEFBSOCON.7SOCENDEFBSOCON.6SOCENDEFBSOCON.10SOCODDEFBSOCON.12SOESDEFBSOCON.12SOLBDEFBSOCON.13SOLBDEFBSOCON.14DEFBSOCON.14DEFBSOCON.14	CC30IE	DEFB	CC30IC.6
CC31IRDEFBCC31IC.7ADCIEDEFBADCIC.6ADCIRDEFBADCIC.7ADEIEDEFBADEIC.6ADEIRDEFBTOIC.6TOIEDEFBTOIC.7TIEDEFBTOIC.7TIEDEFBTTIC.7TRDEFBTTIC.7TRDEFBTTIC.7TRDEFBTTIC.7TRDEFBTTIC.7ADSTDEFBADCON.7ADSTDEFBADCON.7ADSYDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1UNDOPCDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.6SOSTPDEFBSOCON.6SOCENDEFBSOCON.6SOCENDEFBSOCON.7SOPEDEFBSOCON.7SOPEDEFBSOCON.10SOODDDEFBSOCON.12SOBRSDEFBSOCON.14SOLBDEFBSOCON.14SOLBDEFBSOCON.14	CC30IR	DEFB	CC30IC.7
ADCIEDEFBADCIC.6ADCIRDEFBADEIC.7ADEIEDEFBADEIC.6ADEIRDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TILEDEFBTIC.6TIRDEFBTTC.6T7IRDEFBTTC.6T7IRDEFBTTC.7ADSTDEFBTTC.6ADSTDEFBTSIC.6ADSTDEFBADCON.7ADSTDEFBADCON.9ADCINDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.1UNDOPCDEFBTFR.7STKUFDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.1SOSTPDEFBSOCON.1SOSTPDEFBSOCON.1SOFENDEFBSOCON.7SOPENDEFBSOCON.7SOPEDEFBSOCON.7SOPEDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14	CC31IE	DEFB	CC31IC.6
ADCIRDEFBADCIC.7ADEIEDEFBADEIC.6ADEIRDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TIIEDEFBTIIC.6TIRDEFBTIIC.7T7IEDEFBT7IC.6T7IRDEFBT7IC.7T8IEDEFBT8IC.7ADSTDEFBADCON.7ADSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBSOCON.4SOSTPDEFBSOCON.6SOGENDEFBSOCON.7SOPENDEFBSOCON.6SOOENDEFBSOCON.7SOCOENDEFBSOCON.7SOCOENDEFBSOCON.10SOODDDEFBSOCON.12SOBRSDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14	CC31IR	DEFB	CC31IC.7
ADEIEDEFBADEIC.6ADEIRDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TIEDEFBTIC.7TTIEDEFBTTC.7TTIEDEFBTTC.7TRDEFBTTC.7TRDEFBTSIC.7ADSTDEFBADCON.7ADSTDEFBADCON.7ADSTDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBADCON.11ILLBUSDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.6SOSTPDEFBSOCON.6SOGENDEFBSOCON.6SOOENDEFBSOCON.7SOPEDEFBSOCON.7SOPEDEFBSOCON.7SOCOEDEFBSOCON.10SOODDDEFBSOCON.12SOBRSDEFBSOCON.13SOLBDEFBSOCON.14			
ADEIRDEFBADEIC.7TOIEDEFBTOIC.6TOIRDEFBTOIC.7TIIEDEFBTIIC.6TIRDEFBTIC.6TTRDEFBTTC.6T7RDEFBTTC.6T8RDEFBTSC.6ADSTDEFBADCON.7ADSTDEFBADCON.9ADCINDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.4SOSTPDEFBSOCON.5SOFENDEFBSOCON.7SOPENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14	ADCIR		
TOIEDEFBTOIC.6TOIRDEFBTOIC.7T1IEDEFBTIIC.6T1IRDEFBTIIC.7T7IEDEFBT7IC.6T7IRDEFBT7IC.7T8IEDEFBT8IC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCRQDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.13UNDOPCDEFBTFR.13STKOFDEFBTFR.13SOSTPDEFBSOCON.4SOSTPDEFBSOCON.4SOFENDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.12SOFFDEFBSOCON.12SOFFDEFBSOCON.12SOFFDEFBSOCON.12SOEDDEFBSOCON.12SOEDDEFBSOCON.12SOEDDEFBSOCON.12SOEDDEFBSOCON.12SOEDDEFBSOCON.14SOLBDEFBSOCON.14	ADEIE		
TOIRDEFBTOIC.7T1IEDEFBTIIC.6T1IRDEFBTIIC.7T7IEDEFBT7IC.6T7IRDEFBT7IC.7T8IEDEFBT8IC.7ADSTDEFBADCON.7ADSYDEFBADCON.9ADCINDEFBTFR.0ILLINADEFBTFR.0ILLOPADEFBTFR.3UNDOPCDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.15WDTINDEFBSOCON.4SOSTPDEFBSOCON.4SOFENDEFBSOCON.6SOGENDEFBSOCON.7SOFEDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.10SOODDDEFBSOCON.12SOESDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14	ADEIR	DEFB	ADEIC.7
T1IEDEFBT1IC.6T1IRDEFBT1C.7T7IEDEFBT7C.6T7IRDEFBT7IC.7T8IEDEFBT8IC.6T8IRDEFBADCON.7ADSTDEFBADCON.9ADCINDEFBADCON.9ADCRQDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.4SOSTPDEFBSOCON.5SOFENDEFBSOCON.7SOPENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.14DEFBSOCON.14			
T11RDEFBT11C.7T71EDEFBT71C.6T71RDEFBT71C.7BEDEFBT71C.7T81EDEFBT81C.6T81RDEFBT81C.7ADSTDEFBADCON.7ADSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.14DEFBSOCON.12SOLBDEFBSOCON.14			
T7IEDEFBT7IC.6T7IRDEFBT7IC.7T8IEDEFBT8IC.6T8IRDEFBT8IC.7ADSTDEFBADCON.7ADBSYDEFBADCON.9ADCINDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBSOCON.1SOSTPDEFBSOCON.4SOPENDEFBSOCON.4SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.12SOODDDEFBSOCON.12SOLBDEFBSOCON.14SOLBDEFBSOCON.14			
T7IRDEFBT7IC.7T8IEDEFBT8IC.6T8IRDEFBT8IC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBADCON.11ILLBUSDEFBTFR.0ILLINADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.10SOOEDEFBSOCON.10SOODDDEFBSOCON.10SOODDDEFBSOCON.13SOLBDEFBSOCON.14	T1IR		
T&IEDEFBT&IC.6T&IRDEFBT&IC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBADCON.11ILLBUSDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.10SOODDDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.13SOLBDEFBSOCON.14	T7IE	DEFB	
T8IRDEFBT8IC.7ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBADCON.11ILLBUSDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.10SOODDDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.14	T7IR	DEFB	T7IC.7
ADSTDEFBADCON.7ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCRQDEFBADCON.10ADCRQDEFBADCON.11ILLBUSDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.9SOCOEDEFBSOCON.12SODDDEFBSOCON.12SOBRSDEFBSOCON.13SOLBDEFBSOCON.14	T8IE	DEFB	T8IC.6
ADBSYDEFBADCON.8ADWRDEFBADCON.9ADCINDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBADCON.11ILLBUSDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.3UNDOPCDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.8SOFEDEFBSOCON.10SOODDDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.13SOLBDEFBSOCON.14	T8IR	DEFB	T8IC.7
ADWRDEFBADCON.9ADCINDEFBADCON.10ADCRQDEFBADCON.11ILLBUSDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.10SOODDDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.14	ADST		
ADCIN ADCRQDEFB DEFBADCON.10 ADCON.11ILLBUS 	ADBSY	DEFB	
ADCRQDEFBADCON.11ILLBUSDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.7STKUFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.9SOOEDEFBSOCON.12SODDDEFBSOCON.12SOLBDEFBSOCON.14	ADWR	DEFB	
ILLBUSDEFBTFR.0ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.7SOFEDEFBSOCON.7SOFEDEFBSOCON.9SOOEDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.13SOLBDEFBSOCON.14	ADCIN	DEFB	ADCON.10
ILLINADEFBTFR.1ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBWDTCON.0WDTRDEFBWDTCON.1SOSTPDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.9SOOEDEFBSOCON.12SODDDEFBSOCON.12SOLBDEFBSOCON.13	ADCRQ	DEFB	ADCON.11
ILLOPADEFBTFR.2PRTFLTDEFBTFR.3UNDOPCDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOFEDEFBSOCON.7SOPEDEFBSOCON.8SOFEDEFBSOCON.9SOODDEFBSOCON.12SOBRSDEFBSOCON.13SOLBDEFBSOCON.14	ILLBUS		
PRTFLTDEFBTFR.3UNDOPCDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.5SOFENDEFBSOCON.7SOPEDEFBSOCON.9SOCEDEFBSOCON.9SOODDDEFBSOCON.12SOBRSDEFBSOCON.12SOLBDEFBSOCON.13	ILLINA	DEFB	
UNDOPCDEFBTFR.7STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBWDTCON.1SOSTPDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.5SOFEDEFBSOCON.7SOPEDEFBSOCON.9SOOEDEFBSOCON.10SOODDDEFBSOCON.10SOLBDEFBSOCON.13SOLBDEFBSOCON.14	ILLOPA	DEFB	
STKUFDEFBTFR.13STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBWDTCON.1SOSTPDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.9SOOEDEFBSOCON.12SODDDEFBSOCON.13SOLBDEFBSOCON.14	PRTFLT	DEFB	TFR.3
STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBWDTCON.1SOSTPDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOFEDEFBSOCON.7SOFEDEFBSOCON.9SOOEDEFBSOCON.12SODDDEFBSOCON.12SOLBDEFBSOCON.13	UNDOPC	DEFB	TFR.7
STKOFDEFBTFR.14NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBWDTCON.1SOSTPDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.6SOFENDEFBSOCON.6SOFEDEFBSOCON.7SOFEDEFBSOCON.9SOOEDEFBSOCON.12SODDDEFBSOCON.12SOLBDEFBSOCON.13SOLBDEFBSOCON.14	STKUF	DEFB	TFR.13
NMIDEFBTFR.15WDTINDEFBWDTCON.0WDTRDEFBWDTCON.1SOSTPDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.5SOFENDEFBSOCON.7SOPEDEFBSOCON.8SOFEDEFBSOCON.9SOODDEFBSOCON.12SOBRSDEFBSOCON.12SOLBDEFBSOCON.13SOLBDEFBSOCON.14		DEFB	TFR.14
WDTRDEFBWDTCON.1SOSTPDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.5SOFENDEFBSOCON.6SOOENDEFBSOCON.7SOFEDEFBSOCON.8SOFEDEFBSOCON.9SOOEDEFBSOCON.10SODDDEFBSOCON.12SOBRSDEFBSOCON.13SOLBDEFBSOCON.14			TFR.15
SOSTPDEFBSOCON.3SORENDEFBSOCON.4SOPENDEFBSOCON.5SOFENDEFBSOCON.6SOCENDEFBSOCON.7SOPEDEFBSOCON.8SOFEDEFBSOCON.9SOODDEFBSOCON.12SODDDEFBSOCON.13SOLBDEFBSOCON.14	WDTIN	DEFB	WDTCON.0
SORENDEFBSOCON.4SOPENDEFBSOCON.5SOFENDEFBSOCON.6SOOENDEFBSOCON.7SOPEDEFBSOCON.8SOFEDEFBSOCON.9SOOEDEFBSOCON.10SOODDDEFBSOCON.12SOLBDEFBSOCON.13SOLBDEFBSOCON.14	WDTR	DEFB	WDTCON.1
SOPENDEFBSOCON.5SOFENDEFBSOCON.6SOOENDEFBSOCON.7SOPEDEFBSOCON.8SOFEDEFBSOCON.9SOOEDEFBSOCON.10SODDDEFBSOCON.12SOBRSDEFBSOCON.13SOLBDEFBSOCON.14			
SOFENDEFBSOCON.6SOOENDEFBSOCON.7SOPEDEFBSOCON.8SOFEDEFBSOCON.9SOOEDEFBSOCON.10SODDDEFBSOCON.12SOBRSDEFBSOCON.13SOLBDEFBSOCON.14			
SOOENDEFBSOCON.7SOPEDEFBSOCON.8SOFEDEFBSOCON.9SOOEDEFBSOCON.10SOODDDEFBSOCON.12SOBRSDEFBSOCON.13SOLBDEFBSOCON.14	SOPEN	DEFB	
SOPE DEFB SOCON.8 SOFE DEFB SOCON.9 SOOE DEFB SOCON.10 SOODD DEFB SOCON.12 SOBRS DEFB SOCON.13 SOLB DEFB SOCON.14	SOFEN	DEFB	
SOFE DEFB SOCON.9 SOOE DEFB SOCON.10 SOODD DEFB SOCON.12 SOBRS DEFB SOCON.13 SOLB DEFB SOCON.14	SOOEN	DEFB	
SOFE DEFB SOCON.9 SOOE DEFB SOCON.10 SOODD DEFB SOCON.12 SOBRS DEFB SOCON.13 SOLB DEFB SOCON.14	SOPE	DEFB	SOCON.8
S00E DEFB S0CON.10 S00DD DEFB S0CON.12 S0BRS DEFB S0CON.13 S0LB DEFB S0CON.14	SOFE	DEFB	SOCON.9
SOODDDEFBSOCON.12SOBRSDEFBSOCON.13SOLBDEFBSOCON.14		DEFB	SOCON.10
SOBRSDEFBSOCON.13SOLBDEFBSOCON.14			SOCON.12
SOLB DEFB SOCON.14			
SUN DEFE SUCON.IS			
	SUN	DEFD	50000.15

0			
	SSCHB	DEFB	SSCCON.4
	SSCPH	DEFB	SSCCON.5
	SSCPO	DEFB	SSCCON.6
	SSCTEN	DEFB	SSCCON.8
	SSCREN	DEFB	SSCCON.9
	SSCPEN	DEFB	SSCCON.10
	SSCBEN	DEFB	SSCCON.11
	SSCBSY	DEFB	SSCCON.12
	SSCMS	DEFB	SSCCON.14
	SSCEN	DEFB	SSCCON.15
	PTRO	DEFB	PWMCON0.0
	PTR1	DEFB	PWMCON0.1
	PTR2	DEFB	PWMCON0.2
	PTR3	DEFB	PWMCON0.3
	PTIO	DEFB	PWMCON0.4
	PTI1	DEFB	PWMCON0.5
	PTI2	DEFB	PWMCON0.6
	PTI3	DEFB	PWMCON0.7
	PIEO	DEFB	PWMCON0.8
	PIE1	DEFB	PWMCON0.9
	PIE2	DEFB	PWMCON0.10
	PIE3	DEFB	PWMCON0.11
	PIRO	DEFB	PWMCON0.12
	PIR1	DEFB	PWMCON0.13
1	PIR2	DEFB	PWMCON0.14
	PIR3	DEFB	PWMCON0.15
	PENO	DEFB	PWMCON1.0
	PEN1	DEFB	PWMCON1.1
	PEN2	DEFB	PWMCON1.2
	PEN3	DEFB	PWMCON1.3
	PMO	DEFB	PWMCON1.4
	PM1	DEFB	PWMCON1.5
	PM2	DEFB	PWMCON1.6
	PM3	DEFB	PWMCON1.7
	PB01	DEFB	PWMCON1.12
	PS2	DEFB	PWMCON1.14
	PS3	DEFB	PWMCON1.15
	PWMIE	DEFB	PWMIC.6
	PWMIR	DEFB	PWMIC.7
1	XP3IE	DEFB	XP3IC.6
	XP3IR	DEFB	XP3IC.7
	XP2IE	DEFB	XP2IC.6
	XP2IR	DEFB	XP2IC.7
	XP1IE	DEFB	XP1IC.6
	XP1IR	DEFB	XP1IC.7
	XPOIE	DEFB	XPOIC.6
	XPOIR	DEFB	XPOIC.7

B.9 Data Acquisition Node

On the next page starts the code for the Data Acquisiton Node. The files for the node are as follows.

- 1. comp.bat
- 2. main.asm
- 3. cnmod.asm
- 4. canmo.asm
- 5. canint.asm
- 6. timers.asm
- 7. atod.asm
- 8. ema.asm
- 9. linker.lnv
- 10. Reg167b.def

B.10 DC/DC Converter Node

On the next page starts the code for the CAN Router. The files for the node are as follows.

- 1. comp.bat
- 2. main.asm
- 3. cnmod.asm
- 4. canmo.asm
- 5. canint.asm
- 6. linker.lnv
- 7. Reg167b.def

99/05/13 21:51:50

- del *.obj del *.lno
- del *.out
- del *.hex
- a166 main.asm
- al66 timers.asm
- a166 atod.asm
- a166 canmod.asm a166 canmo.asm
- a166 ema.asm
- 1166 LINK main.obj timers.obj atod.obj canmod.obj canmo.obj ema.obj TO main.lno
- 1166 @linker.lnv
- ihex166 -i16 main.out -o main.hex

99/06/12 19:09:02

main.asm



;; End of initialization for the timer that controls the A/D interval times SSEGMENTED SEXTEND ;; Initialize CAN Bus SEXTSFR ; Call the CAN initialization function CALL canin ; CAN USE ALL internal RAM for Stack SEXTSSK ;; End of CAN Bus Initialization SEXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) meto: NOP ; just loop here waiting SSYMBOLS NOP JMP meto NAME main ; define a common register area of 16 register RET ; return RBANK1 COMREG R0-R15 main ENDP mainseg ENDS ; default stack size of 256 Words SSKDEF 4 ; codesegment that contains reset int pointer startupsec SECTION CODE ASSUME DPP3:SYSTEM ; reset interrupt number is zero at Oh sysreset PROC TASK INTNO=0H ; forces next instruction to be located at Oh EXTERN canin: FAR ; Can function ORG 000H ; installs a pointer to the startup routine ; external atod initialization JMP start EXTERN atod initialize:FAR RETI ; return from interrupt EXTERN atod_timer_initialize:FAR sysreset ENDP startupsec ENDS END mainseg SECTION CODE main PROC FAR start: DISWDT ; disable the watchdog timer ; Globally Enable Interrupts both global BSET IEN ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh EINIT ; end initialization ;; End of external memory bus initialization ;; Use Hysteresis for Special Input Thresholds EXTR #1 BSET PICON.1 ;; End of Setting Hysteresis for Special Input Thresholds :: Initialize the Data Page pointers for this section MOV DPP3, #03h ; make DPP3 point to system ;; End of Data Page Pointer Initialization ;; Make the direction of Port 2 to output MOV DP2, ONES ; Pins zero and on are used to capture the direction of BCLR DP2.0 the current flow. BCLR DP2.1 ;; Initialize The Stack :: The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #OFBFFh the highest ;; byte that the stack pointers can point to is #OFBFEh MOV STKUN, #0FBFEh; Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP, #OFBFEh ; Set the Stack Pointer ;; End of Stack Initialization ;; Initialize the Analog to Digital Converter CALL atod_initialize; atod ;; End of A/D initialization ;; Initialize the timer for that controls A/D interval times CALL atod_timer_initialize

99/05/14 19:37:56



canmod.asm

\$SEGMENTED	POP RO
\$EXTEND	RET
\$EXTSFR	canin ENDP
\$EXTMEM	This Pressburg sate all of the Moor ship invalid
\$NOMOD166	setall PROC FAR ; This Procedure sets all of the Mess objs invalid
<pre>\$STDNAMES(reg167b.def)</pre>	;; by using a counter it counts up to 15 and initializes all of the message
\$SYMBOLS	;; objects along the way.
	PUSH R2
NAME canmod	PUSH R4
	PUSH R5
RBANK1 COMREG R0-R15 ; define a common register area of 16 registers	AND R5,ZEROS
GLOBAL canin ; The function must be declared Global at the	OR R5, #Olh ; Set counter to 1 for first MO
; beginning of the module	AND R2,ZEROS
	OR R2,#0EF10h ; Set pointer to MO1
EXTERN canmocfg:FAR ; configures specific Message objects	AND R4, ZEROS
	OR R4, #5555h ; Set R4 to make MObs invalid
ASSUME DPP3:SYSTEM	nextreg:MOV [R2],R4 ; make all message objects invalid
canfunc SECTION CODE ; codesegment that contains reset int pointer	ADD R2, #10h
	CMPI1 R5,#0Ph
canin PROC FAR	JMPA CC_NZ, nextreg ;
PUSH RO	POP R5
PUSH R1	POP R4 POP R2
	RET
;; set all of the CAN control registers	setall ENDP
AND C1CSR,ZEROS ; set control register to zero	Secal ENDP
MOV R1, #0043h ; Set IE and INIT bits	canfunc ENDS
OR C1CSR,R1 ; set control register to R1's value	END
and along any Die bising unicher to some	END
AND C1BTR, ZEROS ; set Bit timing register to zero	
MOV R1, #03447h ; set for 125k operation	
OR C1BTR, R1 ; set Bit timing register parameters	
AND C1GMS, ZEROS ; set Global Mask short register to zero	
MOV R1, #0FFFFh ; EOFF is what DAVE initialize	
OR CIGMS, R1 ; set GMS	
OR CIGHS, RI , SEC GHS	
AND C1UGML, ZEROS ; set Upper global mask long to zero	
MOV R1, #0FFFFh	
OR ClugML, R1	
MOV R1, #0F8FFh	
AND C1LGML, ZEROS	
OR C1LGML, R1 ; lower global mask	
AND C1UMLM, ZEROS	
OR ClUMLM, R1 ; upper mask of last register	
AND CILMLM, ZEROS	
OR C1LMLM, R1 ; lower mask of last register	
CALL setall ; sets all of the CAN registers to off	
CALL setall ; sets all of the CAN registers to off	
CALL canmocfg ; Configures specific Message Objects	
CALL Calmotig , configures specific hessage objects	
;; Setup CAN interrupt and Initialize CAN module	
AND XPOIC, ZEROS ; configure CAN interrupt control Register	
AND ROJZEROS	
OR R0,#0073h ; enable interrupt, level is 10 group is 2	
EXTR #2	
OR XPOIC,R0 ; Configure CAN interrupt Control Register	
BCLR XPOIC.6 ; Turn off interrupts	
AND R1, ZEROS	
OR R1, #00041h ; crashes if I clear the CPU access to the BTR	
XOR CICSR, R1 ; end initialize CAN interrupt	
POP R1	

99/05/23 18:58:22



canmo.asm

<pre>\$SEGMENTED \$EXTSEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS NAME canmo RBANK1 COMREG R0-R15 GLOBAL canmocfg</pre>	; declare bank of 16 global registers	OR R1, #0038h MOV MCD_M2,R1 f data MOV DATA_M2, ZEROS MOV R0, #05595 errupts MOV MCR_M2, R0 ;; This message object is t MOV R2, #MCR_M3 AND R1, ZEROS OR R1, #05555h MOV [R2],R1	<pre>; This is a transmit object ; Databyte(0) = 0 and Set to receive and 3 bytes o ; fill the Data of the MO with Zeros ; This makes a message object valid, but with no int ; Message control Register 2 is now valid he 36v battery temperature message object ; start of Message Object 3 ; set MO3's Control register to inactive</pre>
can_module SECTION CODE		ADD R2,#2h	; point to Upper Arbitration register
ASSUME DPP3:SYSTEM canmocfg PROC FAR PUSH R0 PUSH R1 PUSH R2 PUSH R3 ;; Now set specific CAN	control Registers	AND R3, ZEROS OR R3, #0007h MOV [R2],R3 ADD R2, #2h MOV [R2], ZEROS AND R1, ZEROS OR R1, #0038h MOV MCD_M3,R1	<pre>; set R3 to zero ; message id for message object 3 ; message id = #0007h ; Point to the Lower Arbitration Register ; standard Message object so lowerarb = 0h ; This guy is a transmit object ; Databyte(0) = 0 and Set to receive and 3 bytes o</pre>
;; initialize message ob ;; initializing this ob ;; the comment "Setup CA		f data MOV DATA_M3, ZEROS MOV R0, #05595 errupts MOV MCR_M3, R0	; fill the Data of the MO with Zeros ; This makes a message object valid, but with no int ; Message control Register 3 is now valid
;; This message object i n if ;; it is requested by ar	is the 36v battery voltage and should send the informatio nother node		state of charge message object the state of charge at the request of another message ob
MOV R2, #MCR_M1 AND R1, ZEROS OR R1, #5555h perating on it MOV [R2],R1 ; set MO	; start of Message Object 1 ; Make sure that this message object is invalid before o D1's Control register	MOV R2, #MCR_M4 AND R1, ZEROS OR R1, #05555h	e other message objects because it has a data length of 5 ; start of Message Object 4 set MO2's Control register
ADD R2,#2h	; point to Upper Arbitration register		
AND R3, ZEROS OR R3, #0005h MOV [R2],R3 ADD R2, #2h MOV [R2], ZEROS AND R1, ZEROS OR R1, #0038h MOV MCD_M1,R1 MOV DATA_M1, ZEROS MOV R0, #05595	<pre>; set R3 to ; message id for message object 1 ; message id = #0005h ; Point to the Lower Arbitration Register ; standard Message object so lowerarb = 0h ; put 0AAh into first data byte and set to transmit ; Databyte(0) = 0 and Set to receive and 3 bytes of data ; fill the Data of the MO with Zeros ; This makes a message object valid, but with no interru</pre>	ADD R2,#2h AND R3, ZEROS OR R3, #0008h MOV [R2],R3 ADD R2, #2h MOV [R2], ZEROS AND R1, ZEROS OR R1, #0058h MOV MCD_M4,R1 data	<pre>; point to Upper Arbitration register ; set R3 to ; message id for message object 4 ; message id = #0009h ; Point to the Lower Arbitration Register ; standard Message object so lowerarb = 0h ; This guy is a transmit object ; Databyte(0) = 0 and Set to receive and 3 bytes of</pre>
pts MOV MCR_M1, R0	; Message control Register 1 is now valid	MOV DATA_M41, ZEROS MOV DATA_M42, ZEROS MOV R0, #05595	
;; it is set up to tran: MOV R2, #MCR_M2 AND R1, ZEROS OR R1, #05555h	is the 36v battery current and direction information smit the information if it is requested by another node ; start of Message Object 2 02's Control register	errupts MOV MCR_M4, R0 ;; This is the 12v battery	; Message control Register 4 is now valid y voltage message object ge object with data length of 3 ; start of Message Object 5
ADD R2,#2h AND R3, ZEROS OR R3, #0006h MOV [R2],R3 ADD R2, #2h MOV [R2], ZEROS AND R1, ZEROS	<pre>; point to Upper Arbitration register ; set R3 to ; message id for message object 2 ; message id = #0006h ; Point to the Lower Arbitration Register ; standard Message object so lowerarb = 0h</pre>	OR R1, #05555h	set M05's Control register ; point to Upper Arbitration register ; set R3 to ; message id for message object 5 ; message id = #0009h

99/05/23 18:58:22





; Point to the Lower Arbitration Register ADD R2, #2h MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS ; This guy is a transmit object OR R1, #0038h ; Databyte(0) = 0 and Set to receive and 3 bytes of data MOV MCD M5, R1 ; fill the Data of the MO with Zeros data MOV DATA_M5, ZEROS ; This makes a message object valid, but with no interru MOV R0, #05595 pts : Message control Register 5 is now valid MOV MCR M5, R0 ;; This is the 12v battery current and direction message object ;; it will transmit this information at the request of a remote from MOV R2, #MCR_M6 ; start of Message Object 6 AND R1, ZEROS ; Generate a Receive Interrupt if this message object ac OR R1, #05555h tivates ; set MO6's Control register MOV [R2], R1 ; point to Upper Arbitration register ADD R2,#2h AND R3, ZEROS ; set R3 to OR R3, #000BAh ; message id for message object 6 MOV [R2], R3 ; message id = #000Ah ; Point to the Lower Arbitration Register ADD R2, #2h ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; This guy is a transmit object OR R1, #0038h : Databyte(0) = 0 and Set to receive and 3 bytes of data MOV MCD_M6,R1 ; fill the Data of the MO with Zeros MOV DATA_M6, ZEROS data MOV R0, #05595 ; This makes a message object valid, but with no interru pts ; Message control Register 6 is now valid MOV MCR_M6, R0 ;; This is the 12v battery temperature message object ;; It is setup to transmit the temperature information if an appropriate remote from is received ; start of Message Object 7 MOV R2, #MCR_M7 AND R1, ZEROS OR R1, #05555h : set MO7's Control register MOV [R2], R1 ADD R2,#2h ; point to Upper Arbitration register END ; set R3 to AND R3, ZEROS ; message id for message object 7 OR R3, #000Bh ; message id = #000Bh MOV [R2], R3 ; Point to the Lower Arbitration Register ADD R2, #2h ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; This is a transmit object with 3 data bytes OR R1, #0038h ; Databyte(0) = 0 and Set to receive and 3 bytes of data MOV MCD_M7,R1 ; fill the Data of the MO with Zeros MOV DATA_M7, ZEROS ; This makes a message object valid, but with no interru MOV R0, #05595 pts ; Message control Register 7 is now valid MOV MCR M7, R0 :: This message object contains the 12v battery state of charge. ;; It is similar to message object 4 in that it is setup to transmit 5 data bytes ; start of Message Object 8 MOV R2, #MCR_M8 AND R1, ZEROS OR R1, #05555h ; set MO8's Control register MOV [R2], R1 ; point to Upper Arbitration register ADD R2, #2h AND R3. ZEROS ; set R3 to OR R3, #0000Ch ; message id for message object 8 ; message id = #000Ch MOV [R2], R3

: Point to the Lower Arbitration Register ADD R2, #2h ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; This guy is a transmit object OR R1, #0038h ; Databyte(0) = 0 and Set to receive and 3 bytes of MOV MCD_M8,R1 ; fill the Data of the MO with Zeros MOV DATA M81, ZEROS ; fill the Data of the MO with Zeros MOV DATA_M82, ZEROS : This makes a message object valid, but with no int MOV R0, #05595 errupts MOV MCR_M8, R0 ; Message control Register 8 is now valid ;; This message object is set up to transmit the state of the DC/DC converter ;; The state of the DC/DC converter is the output of the Energy Management algorithm ; start of Message Object 9 MOV R2, #MCR_M9 AND R1, ZEROS OR R1, #05555h ; set MO2's Control register MOV [R2], R1 ADD R2, #2h ; point to Upper Arbitration register AND R3, ZEROS ; set R3 to OR R3, #0000Eh ; message id for message object 8 MOV [R21, R3 ; message id = #000Ch ADD R2, #2h ; Point to the Lower Arbitration Register ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS OR R1, #0038h ; This guy is a transmit object MOV MCD_M9,R1 ; Databyte(0) = 0 and Set to receive and 3 bytes of ; fill the Data of the MO with Zeros MOV DATA M9. ZEROS MOV R0, #05595 ; This makes a message object valid, but with no int errupts ; Message control Register 9 is now valid MOV MCR_M9, R0 POP R3 POP R2 POP R1 POP RO RET canmocfg ENDP can module ENDS

99/05/13 22:24:18

timers.asm

\$SEGMENTED ; These are assembler controls SEXTEND SEXTSFR \$EXTMEM \$EXTINSTR \$NOMOD166 \$STDNAMES(reg167b.def) ; Assembler controls end here \$SYMBOLS NAME timer_functions ASSUME DPP3:SYSTEM RBANK1 COMREG R0-R15 GLOBAL timer_period GLOBAL atod_timer_initialize atod_timer_data SECTION DATA WORD GLOBAL 'ROM' timer_period DW 04990h ; This value plus the time necessary for all conversions is about 1 second atod_timer_data ENDS atod_timer SECTION CODE atod_timer_initialize PROC FAR PUSH DPP0 MOV DPPO, #PAG atod_timer_data MOV T3CON, #0086h ; setup Core Timer T3 for count down mode ; Interrupt stuff MOV T3IC, #002Bh ; enable the interrupt BSET T3IE ; This value plus the time for all conversions i MOV T3, DPP0:timer_period s 1 second BSET T3CON.6 POP DPP0 RET atod timer_initialize ENDP atod_interrupt PROC TASK INTNO=023h ORG 08Ch CALL atod_timer_handler RETI atod_interrupt ENDP atod_timer_handler PROC FAR PUSH DPP0 PUSH RO MOV DPP0, #PAG atod_timer_data ; stop the timer BCLR T3R MOV T3, DPP0:timer_period ; Reset the count down register ; start an A/D conversion BSET ADST POP R0 POP DPP0 RET atod_timer_handler ENDP atod_timer ENDS END

99/06/13 20:28:56

\$SEGMENTED SEXTEND \$EXTSFR ; CAN USE ALL internal RAM for Stack SEXTSSK SEXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) SSYMBOLS name atod ASSUME DPP3:SYSTEM RBANK1 COMREG R0-R15 EXTERN energy management algorithm: FAR GLOBAL atod initialize GLOBAL voltage_36v GLOBAL current 36v GLOBAL current_direction_36v GLOBAL temperature_36v GLOBAL soc_36v_high_word GLOBAL soc_36v_low_word GLOBAL voltage 12v GLOBAL current_12v GLOBAL current direction 12v GLOBAL temperature_12v GLOBAL soc_12v_high_word GLOBAL soc_12v_low_word GLOBAL soc_region_36v GLOBAL soc_region_12v GLOBAL r1_soc_36v_high GLOBAL r1_soc_36v_low GLOBAL r2 soc 36v high GLOBAL r2 soc 36v_low GLOBAL r3_soc_36v_high GLOBAL r3 soc 36v low GLOBAL r4 soc_36v_high GLOBAL r4_soc_36v_low GLOBAL r1_soc_12v_high GLOBAL r1 soc 12v_low GLOBAL r2_soc_12v_high GLOBAL r2_soc_12v_low GLOBAL r3 soc 12v_high GLOBAL r3_soc_12v_low GLOBAL r4_soc_12v_high GLOBAL r4 soc 12v low ;; This A/D is set up to measure the current in two different ;; loads. Because this software is to be used as part of ;; 42volt bus node 1, it uses the names of the loads that ;; that node is supposed to control. ;; The analog to digital converter uses Port 5 atod_data_section SECTION DATA WORD GLOBAL 'RAM' voltage_36v DSW 1 current 36v DSW 1 current direction_36v DSW 1 ; Collected, but not used because no sensor is h temperature_36v DSW 1 ooked up ; The 36v Battery STATE of charg soc_36v_high_word DSW 1 soc_36v_low_word DSW 1 soc_region_36v DSW 1 ; This is the SOC Region (1->5) in which the Battery is

atod.asm

operating voltage_12v DSW 1 current 12v DSW 1 current_direction_12v DSW 1 ; collected, but not used because no sensor temperature 12v DSW 1 is hooked up 5/5/99 : The 12v Battery STATE of c soc_12v_high_word DSW 1 harge soc_12v_low_word DSW 1 soc_region_12v DSW 1 ; This is the SOC Region (1->5) in which the Battery is operating ;; These variables help with the computation total_period DSW 1 atod data section ENDS battery_model_parameters SECTION DATA WORD GLOBAL 'ROM' 063E6h starting_charge_36v_low DW starting_charge_36v_high DW 010h 076A0h starting_charge_12v_low DW 025h starting_charge_12v_high DW rl soc 36v_high DW 012h r1_soc_36v_low DW 07A0h r2_soc_36v_high DW 011h r2_soc_36v_low DW 035DCh r3_soc_36v_high DW 0Dh r3_soc_36v_low DW 0EE86h r4_soc_36v_high DW 09h r4_soc_36v_low DW 0D58Ah r1_soc_12v_high DW 029h r1_soc_12v_low DW 0359Ch r2 soc 12v high DW 027h r2_soc_12v_low DW 05650h r3_soc_12v_high DW 01Fh r3_soc_12v_low DW 0D7F4h r4_soc_12v_high DW 016h r4_soc_12v_low DW 07A4Ch battery_model_parameters ENDS atod setup SECTION CODE atod initialize PROC FAR ;; Initialize variables PUSH DPP0 PUSH DPP1 PUSH DPP2 PUSH RO PUSH R1 PUSH R2 PUSH R3 ;; This section of code simply clears all of the variables which are to be u sed during ;; data collection. ;; It also initializes the amphours of each of the batteries :: The idea is that the system will boot up thinking that both of the batter ies are okav ;; Then it will take and measure the voltages and determine from a graph whi ch is figure xxx ;; in the master's thesis of James Geraci, what the actual state of charge i s. MOV DPP0, #PAG atod_data_section MOV DPP1, #PAG battery_model_parameters AND DPP0:voltage_36v, ZEROS MOV R0, DPP1:starting_charge_36v_low



20.28.56

harge collected

g to make the total

POP R3

POP R2 POP R1

POP RO

POP DPP2 POP DPP1

POP DPP0

RET

atod setup ENDS

atod initialize ENDP

atod function PROC FAR ;; this function works by seeing if the converter is converting

:; for the heater_measurement. If the bit is set, then

;; the bit gets cleared and the IP jumps to where the

;; value in the converter is moved into the heater_current

:: variable.



;; otherwise the bit gets set and the value is moved into ;; the heated_rear_window_current variable ;; The Order of Conversion is: ;; 1) 36v temperature ;; 2) 12v temperature ;; The channels of the A/D are ;; 4) 12v temperature ;; 5) 36v temperature MOV DPPO, #PAG atod data section MOV DPP1, #PAG battery_model_parameters MOV R0, ADDAT ; Get the information from the A/D converter AND R1, ZEROS ; Clear R1 ; The upper nibble of the upper byte of the A/D info rmation gives channel information SHR R1. #04h ; Shift R1 right one nibble. this puts the converter number ; Make a copy of the current information ;; This piece of code isolates the DATA that has just been collected AND RO, #03FFh ; This makes the upper 6 bytes zero ;; This code decides which piece of information has just been collected ;; and goes to the appropriate handler routine CMPB RL1. #05h : This tests to see if the conversion that just finished was JMP cc Z, temperature_36v_routine ; Converter number 5 should take in the temperature for the 36v battery CMPB RL1, #04h ; This tests to see if the conversion that just finished was ; Converter number 4 should take in JMP cc_z, temperature_12v_routine the temperature for the 12v battery NOP CMPB RL1, #03h ; This tests to see if the conversion that just finished was made by converter number 3

JMP cc_z, voltage_36v_routine ; Converter number 3 should take in the volt age for the 36v battery

99/06/13 20:28:56

atod.asm



NOP CMPB RL1, #02h ; This tests to see if the conversion that just finished was mad e by converter number 2 JMP cc_z, voltage_12v_routine ; Converter number 2 should take in the voltage for the 12v battery NOP CMPB RL1, #01h ; This tests to see if the conversion that just finished was mad e by converter number 1 JMP cc_z, current_12v_routine ; Converter number 1 should take in the current for the 12v battery C NOP CMPB RL1, #00h ; This tests to see if the conversion that just finished was mad e by converter number 0 JMP cc_z, current_36v_routine ; Converter number 0 should take in the current for the 36v battery NOP temperature_36v_routine: ;; The information for the temp of 36v battery goes into CAN MO 3 ; This bit pattern deactives MCRs MOV R2, #05555h ; SAVE the Configuration of the MCR MOV R1, #05595h : Turn Off the Message Control Register MOV MCR M3, R2 MOV DATA_M3, R0 ; Put the Data that has just been collected into Message Object 3 ;put the data into memory MOV DPP0:temperature_36v, R0 MOV MCR M3, R1 JMP exit_routine temperature_12v_routine: ;; The information for the temp of 12v battery goes into CAN MO 7 ; This bit pattern deactives MCRs MOV R2, #05555h : SAVE the Configuration of the MCR MOV R1, #05595h ; Turn Off the Message Control Register MOV MCR_M7, R2 MOV DATA_M7, R0 ; Put the Data that has just been collected into Message Object 7 ; Put the 12v temperature into memory MOV DPP0:temperature_12v, R0 MOV MCR M7, R1 JMP exit_routine voltage_36v_routine: ;; The information for the voltage of 36v battery goes into CAN MO 1 ; This bit pattern deactives MCRs MOV R2, #05555h ; SAVE the Configuration of the MCR MOV R1, #05595h ; Turn Off the Message Control Regis MOV MCR M1, R2 MOV DATA_M1, R0 ; Put the Data that has just been collected into Message Object 1 MOV MCR_M1, R1 JMP exit_routine voltage_12v_routine: ;; The information for the voltage of 12v battery goes into CAN MO 5 ; This bit pattern deactives MCRs MOV R2, #05555h : SAVE the Configuration of the MCR MOV R1, #05595h ; Turn Off the Message Control Register MOV MCR_M5, R2 MOV DATA_M5, R0 ; Put the Data that has just been collected into Message Object 5 MOV MCR_M5, R1 JMP exit_routine current_12v_routine:

;; The information for the current of the 12v battery goes into CAN MO 6 ; This bit pattern deactives MCRs MOV R2, #05555h ; SAVE the Configuration of the MCR MOV R1, #05595h ; Turn Off the Message Control Register MOV MCR_M6, R2 ; SAVE the configuration for MCR8 which is the 12v S MOV R8, #05595h OC message object ; Turn off MC8 MOV MCR M8, R2 ;; The State of Charge of the Battery is also generated Here :: The current measurement must be converted back into the actual cu rrent value MOV R3, DPP0:soc_12v_low_word ; The Low byte of the 12v battery so MOV R4, DPP0:soc_12v_high_word ; The upper byte of the 12v battery SOC :: Now we must check to see if the charge is positive or negative ;; This can be done for the 12v battery by checking to see if pin P2 .1 is a one or a zero MOV R2, P2 AND R2, #0002h ; This isolates the pin P2.1 CMP R2, #0002h ; This performs the comparison and sets the Z condit ion flag JMP cc_NZ, perform_addition ; The Pin is brought Low when the Battery is charging ; The battery is discharging perform subtraction: SUB R3, R0 SUBC R4, ZEROS JMP continue data collection ; The battery is charging perform_addition: ADD R3, R0 ADDC R4, ZEROS ;; When this point is reached the SOC should be in registers R3 and R4. The total charge for this period ;; should be in R0, the current direction should be in R2, and the c urrent magnitude should be in R7 continue data collection: ; Put the current into memory MOV DPP0:current 12v, R0 MOV DPP0:current_direction_12v, R2 ; Put the current direction into memory MOV DPP0:soc_12v_high_word, R4 ; Put the upper part of the SOC into memory ; Put the lower part of the SOC into MOV DPP0:soc_12v_low_word, R3 memory MOVB RH2, RL2 ; Move the current direction into the upper byte of R2 AND R2, #00F00h ; Get rid of all but the 3rd nibble SHL R2, #04h ; Move the direction information into the upper nibble : Move the magnitude of the current into R2 ADD R2. R0 MOV DATA M6, R2 ; Put the Data that has just been collected into Message Obj ect 6 ;; These lines put the SOC into the CAN message object number 8 MOV DATA_M81, R4 ; Put the high data byte into data registers 2 and 1 ; Put the low data byte into data registers 4 and 3 MOV DATA_M82, R3 MOV MCR_M8, R8 ; Restore the SOC Message Object MOV MCR_M6, R1 ; Restore the CAN message object to operational status JMP exit routine current 36v routine: ;; The information for the current of the 12v battery goes into CAN MO 6 ; This bit pattern deactives MCRs MOV R2, #05555h ; SAVE the Configuration of the MCR MOV R1, #05595h ; Turn Off the Message Control Register for message MOV MCR_M2, R2

99/06/13 20:28:56

2015	(1175)			
		74		

POP R1 object 2 POP RO ; Turn off MCR4 MOV MCR M4, R2 POP DPP2 POP DPP1 ;; The State of Charge of the Battery is also generated Here ;; The current measurement must be converted back into the actual curren POP DPP0 RET t value MOV R3, DPP0:soc_36v_low_word ; The Low byte of the 36v battery soc atod_function ENDP MOV R4, DPP0:soc_36v_high_word ; The upper byte of the 36v battery soc atod_handlers ENDS ;; Now we must check to see if the charge is positive or negative END ;; This can be done for the 36v battery by checking to see if pin P2.0 i s a one or a zero MOV R2, P2 AND R2, #0001h ; This isolates the pin P2.0 CMP R2, #0001h ; This performs the comparison and sets the Z condition flag JMP cc_NZ, perform_addition_36v ; The battery is charging when the pin is logic level low perform_subtraction_36v: ; The battery is discharging SUB R3, R0 SUBC R4, ZEROS JMP continue_data_collection_36v ; the battery is charging perform_addition_36v: ADD R3, R0 ADDC R4, ZEROS ;; When this point is reached the SOC should be in registers R3 and R4. The total charge for this period ;; should be in R0, the current direction should be in R2, and the curre nt magnitude should be in R7 continue_data_collection_36v: MOV DPP0:current_36v, R0 MOV DPP0:current_direction_36v, R2 MOV DPP0:soc_36v_high_word, R4 MOV DPP0:soc_36v_low_word, R3 MOVB RH2, RL2 ; Move the current direction into the upper byte of R2 ; Get rid of all but the 3rd nibble AND R2, #00F00h SHL R2, #04h ; Move the direction information into the upper nibble ; Move the magnitude of the current into R2 ADD R2, R0 MOV DATA_M2, R2 ; Magnitude and direction information is now put into Message Ob ject 2 ; Move the SOC into the Message Object 4 ; Put the high data byte into data registers 2 and 1 MOV DATA M41, R4 ; Put the low data byte into data registers 4 and 3 MOV DATA M42, R3 MOV MCR_M4, R1 ; Restore the SOC Message Object MOV MCR_M2, R1 ; Restore the CAN message object to operational status CALL energy_management_algorithm MOV R9, #04h ADD P2, R9 ; Start the Conversion Again BSET T3R JMP exit_routine exit routine: POP R9 POP R8 POP R7 POP R6 POP R5 POP R4 POP R3 POP R2

99/06/14 05:10:20

					1		
urposes	real	value	is	2			

\$SEGMENTED	decision_11_pm		
SEXTEND	decision_11_pp	DW 2	
\$EXTSFR	decision_12_mm	DM 0	
\$EXTSSK ; CAN USE ALL internal RAM for Stack			
SEXTMEM	decision_12_mp	DW 3	
SNOMOD166	decision_12_pm	DW 0	
	decision_12_pp		
\$STDNAMES(reg167b.def)	decision_12_pp	Dir 4	
\$SYMBOLS		0.000	
HIGNEY OLDER HIGH	decision_13_mm	DW 2	
	decision_13_mp	DW 0	
name ema2 ; THIS IS THE ENERGY MANAGEMENT ALGORITHM ASSEMBLY FILE	decision_13_pm		;modified for test purposes real value is
name ema2 ; THIS IS THE ENERGY MANAGEMENT ALGORITHM ASSEMBLY FILE	decision_13_pp		/modified for for provention
	decision_13_pp	DW Z	
ASSUME DPP3:SYSTEM			
RBANK1 COMREG R0-R15	decision_14_mm	DW 2	
GLOBAL energy_management_algorithm	decision_14_mp	DW 2	
	decision_14_pm		
GLOBAL dcdcinitialize	decision_14_pp		
	decision_14_pp	DW 2	
EXTERN voltage_36v:WORD			
EXTERN current_36v:WORD	decision_15_mm	DW 2	
EXTERN current_direction_36v:WORD	decision_15_mp	DW 2	
	decision_15_pm		
EXTERN temperature_36v:WORD	decision_15_pp		
EXTERN soc_36v_high_word:WORD	decision_is_pp	DW Z	
EXTERN soc_36v_low_word:WORD			
EXTERN voltage_12v:WORD	decision_21_mm	DW 3	
	decision_21_mp	DW 1	
EXTERN current_12v:WORD	decision_21_pm		
EXTERN current_direction_12v:WORD			
EXTERN temperature_12v:WORD	decision_21_pp	DW 1	
EXTERN soc_12v_high_word:WORD			
	decision_22_mm	DW 0	
EXTERN soc_12v_low_word:WORD	decision_22_mp		
EXTERN soc_region_36v:WORD			
EXTERN soc_region_12v:WORD	decision_22_pm		
	decision_22_pp	DW 2	
EXTERN r1_soc_36v_high:WORD			
	decision_23_mm	DW 0	
EXTERN r1_soc_36v_low:WORD			
EXTERN r2_soc_36v_high:WORD	decision_23_mp		
EXTERN r2_soc_36v_low:WORD	decision_23_pm	DW 4	
EXTERN r3_soc_36v_high:WORD	decision_23_pp	DW 4	
EXTERN r3_soc_36v_low:WORD	decision_24_mm	DIA 2	
EXTERN r4_soc_36v_high:WORD			
EXTERN r4_soc_36v_low:WORD	decision_24_mp		
	decision_24_pm	DW 2	
EXTERN r1_soc_12v_high:WORD	decision_24_pp	DW 2	
EXTERN r1_soc_12v_low:WORD	de sister DE ser	DM 0	
EXTERN r2_soc_12v_high:WORD	decision_25_mm		
EXTERN r2_soc_12v_low:WORD	decision_25_mp	DW 2	
EXTERN r3_soc_12v_high:WORD	decision_25_pm	DW 2	
	decision_25_pp	DW 2	
EXTERN r3_soc_12v_low:WORD			
EXTERN r4_soc_12v_high:WORD	a 24	DIA 1	
EXTERN r4_soc_12v_low:WORD	decision_31_mm		
	decision_31_mp	DW 1	
dcdc_data_section SECTION DATA WORD GLOBAL 'RAM'	decision_31_pm	DW 1	
	decision_31_pp	DW 1	
dcdc_state DSW 1	deerbron_br_pp	211 1	
dcdc_data_section ENDS			
	decision_32_mm	DW 1	
dcdc_decisions SECTION DATA WORD GLOBAL 'ROM'	decision_32_mp	DW 1	
;; There are 5 decisions to be made	decision_32_pm	DW 1	
	decision_32_pp		
;; 0 = NONE	l recipion_32_pp	D11 1	
;; 1 = Full	a to the second second	1000	
;; 2 = ZERO	decision_33_mm	DW 0	
	decision_33_mp	DW 0	
;; 3 = UP	decision_33_pm		
;; 4 = DOWN			
;; The hex symbol next to some of the values is unnecessary but was put	decision_33_pp	DW U	
:: in for test purposes			
	decision_34_mm	DW 4	
decision_11_mm DW 0	decision_34_mp		
decision_11_mp DW 1	Geoiston_24_mb	211 1	

99/06/14 05:10:20

	decision	n_34_pm	DW	4	
	decision	n_34_pp	DW	4	
	decision	n_35_mm	DW	2	
			DW		
	decision	n_35_pm			
	decision	n 35 pm	DW	2	
	decisio.	11_33_pp	Di	2	
	decision	n_41_mm	DW	1	
	decision	n_41_mp	DW	1	
	decision	n_41_pm	DW	1	
	decision	n_41_pp	DW	1	
		10	DU	1	
	decisio	n_42_mm	DW		
			DW		
			DW		
	decisio	n_42_pp	DW	1	
	decisio	n_43_mm	DW	3	
			DW		
			DW		
		n_43_pp	DW		
	uecisio.	11_42_pp	DW	5	
	decisio	n_44_mm	DW	4	
	decisio	n_44_mp	DW	3	
		n_44_pm	DW	4	
		n_44_pp	DW	0	
		45	D (1)	2	
			DW		
			DW		
			DW		
	decisio	n_45_pp	DW	2	
	decisio	n 51 mm	DW	2	
	decisio	n_51_mm n_51_mp	DW	2	
			DW		
	decisio		DW		
				2	
	decisio	n_52_mm	DW	2	
	decisio	n_52_mp	DW	2	
	decisio	n_52_pm	DW	2	
	decisio	n_52_pp	DW	2	
				0	
	decisio	n_53_mm n_53_mp	DW	2	
	decisio	n_53_mp	DW	2	
	decisio	n_53_pm	DW		
	decisio	n_53_pp	DW	2	
	decisio	n 54 mm	DW	4	
	decisio	n_54_mm n_54_mp	DW	3	
			DW		
		n_54_pm	DW		
	decisio	n_55_mm n_55_mp	DW	0	
	decisio	n_55_mp	DW	0	
	decisio	n_55_pm	DW	0	
			DW	0	
dcdc_ded					
1 . 1		ON CODE			
dcdcsta					
acacini		PROC FAR		imple.	ini+-
	;; This	function	II S.	TWDTA	111101

;; This function simply initializes the DC/DC converter to ZERO output FUSH DPPO MOV DPPO, #PAG dcdc_data_section NOP

ema.asm



MOV DPP0:dcdc state, ZEROS POP DPP0 RET dcdcinitialize ENDP dcdcstart ENDS energy_management SECTION CODE energy_management_algorithm PROC FAR PUSH RO PUSH DPP0 CALL determine_soc_36v CALL determine_soc_12v CALL ema_decision MOV DPP0, #PAG dcdc_data_section NOP MOV R0, DPP0:dcdc_state MOV DATA_M9, R0 MOV RO, #06595h ; transmit the data in DATA_M9 which happens to be the wante d DC/DC converter state MOV MCR M9, R0 POP DPP0 POP RO RET energy_management_algorithm ENDP energy_management ENDS energy_management_options SECTION CODE ema decision PROC FAR :: This function takes and makes a decision as to what to do about the state of the DC/DC converter ;; Based on the Region of state of charge of both batteries and their currents ;; It does this by using a giant WORD lookup table. This WORD is put into the varia ble ;; dcdcstate1, and from there it is decided what to do with it. PUSH RO PUSH R1 PUSH R2 PUSH R3 PUSH R4 PUSH R5 PUSH R6 PUSH R7 PUSH R8 PUSH R9 PUSH R10 PUSH R11 PUSH MDH PUSH MDL PUSH DPP0 PUSH DPP1 PUSH DPP2 MOV DPPO, #PAG current_direction_36v AND R6, ZEROS ; This is to be used in looking up the array index. AND R7, ZEROS ; This is to be used as a pointer to our array. ;; These are the variables needed to make a decision about the ;; State of the DC/DC converter MOV R0, DPP0:current_direction_36v MOV R1, DPP0:current_direction_12v MOV R2, DPP0:soc_region_36v MOV R3, DPP0:soc region 12v ;; The function for computing the memory location to look in is ;; The soc_region_12v - 1 = the number of 20s in the offset ;; The soc_region_36v -1 = the number of 4s in the offset ;; and the current signs gives one of 4 different offsets ;; $(12,36) \Rightarrow (-,-) = 1$; (-, +) = 2; (+,-) = 3; (+,+) = 4;; Adding them all togther gives you up to 100 different choices







;; Subtracting by one gives the appropriate array index :: First determine the number of 20s SUB R3, #01h ; R3 now contains the number of 20s that are in offset index ;; Now determine the number of 4s in the index SUB R2, #01h ; R2, now contains the number of 4s that are in the index. :: Now Compute the Major index by unsigned multiplication MOV R4, #14h MULU R3, R4 ; 14h is 20 in hex NOP ; Now R3 contains a number between zero and 80 MOV R3, MDL MOV R4, #4h : 4h is 4 in hex MULU R2. R4 NOP ; Now R2 contains a number between zero and 16 MOV R2, MDL NOP ; Now R3 has the index less the offset of 4 created by t ADD R3, R2 he current signs. ;; Now Determine the offset due to the current direction. CMP R1, ZEROS ; Test the 12v current direction JMP cc_Z, plus_12v minus 12v: CMP R0, ZEROS ; Test the 36v current direction JMP cc_Z, plus_one_36v ; Negative 36v current direction MOV R5, #01h JMP finalize index plus_12v: CMP R0, ZEROS ; test the 36v current direction JMP cc_Z, plus_two_36v MOV R5, #03h JMP finalize_index plus_one_36v: MOV R5, #02h JMP finalize_index plus_two_36v: MOV R5, #04h finalize index: MOV R0, #02h ADD R3, R5 MULU R3, RO NOP MOV R3, MDL SUB R3, #02h ;; Now R3 has the final index. Now the appropriate word can be loo ked up in our lookup table. MOV DPP2, #PAG dcdc_decisions NOP ; move the address of the first item in the arra MOV R8, #DPP2:dcdc_decisions y into register 8 ADD R8, R3 get_data: MOV R9, [R8] ; This puts the decision of the DC/DC converter into R9 MOV DPP1, #PAG dcdc_data_section NOP MOV DPP1:dcdc_state, R9 ;; Finally test the 12v battery's voltage ;; if it is less than 13v Go to full on

MOV DPPO, #PAG voltage_12v NOP MOV R10, DPP0:voltage_12v MOV R11, #03FFh CMP R11, R10 JMP cc_NC, full_on CMP R9, ZEROS ; In this case don't do anything JMP cc_Z, exit_dcdc_index CMP R9, #01h ; full on JMP cc_Z, full_on CMP R9, #02h ; Full off JMP cc_Z, full_off CMP R9, #03h ; Up one JMP cc_Z, up_one CMP R9, #04h ; Down one JMP cc_Z, down_one JMP exit_dcdc_index full on: MOV DPP1, #PAG dcdc_data_section NOP : ZEROS produces full on for the DC/DC conve MOV DPP1:dcdc_state, ZEROS rter JMP exit_dcdc_index full_off: MOV DPP1, #PAG dcdc_data_section NOP ; ONES produces full off for the DC/DC conve MOV DPP1:dcdc state, ONES rter JMP exit_dcdc_index up_one: MOV DPP1, #PAG dcdc_data_section NOP MOV R0, DPP1:dcdc_state CMPB RLO, #000h ; see if already at max JMP cc_Z, exit_dcdc_index SUB R0, #01h ; New value for the DC/DC converter MOV DPP1:dcdc state, R0 JMP exit dcdc index down_one: MOV DPP1, #PAG dcdc_data_section NOP MOV R0, DPP1:dcdc_state CMPB RLO, #OFFh ; see if already at min JMP cc_Z, exit_dcdc_index ADD R0, #01h ; new value for DCDC converter MOV DPP1:dcdc state, R0 JMP exit dcdc index exit_dcdc_index: POP DPP2 POP DPP1 POP DPP0 POP MDL POP MDH POP R11 POP R10 POP R9 POP R8 POP R7

99/06 05:10		asm	4
	DP R6 OP R5 OP R4 OP R3 OP R2 OP R1	n l	<pre>; than r2_soc_36v_high so a carry was generated ; soc_36v_high_word > r2_soc_36v_high => Very Dangerous Over Charge => Regio JMP cc_C, Region2_36v ; If no Carry must test to see if soc_36v_high_word = r2_soc_36v_high</pre>
PO RI ema_decis:	OP RO ET		; If they DON'T equal then soc_36v_high_word < r2_soc_36v_high ; This means Test for Different Region JMP cc_NZ, Test_Region_3_36v
determine	_soc_region SECTION CODE		<pre>; Since soc_36v_high_word = r2_soc_36v_high must now test lower word ; Inorder to determine if battery is in region 2 or region 3 MOV R0, DPP0:soc_36v_low_word MOV R1, DPP1:r2_soc_36v_low CMP R1, R0 ; This subtracts soc_36v_low_word from r2_soc_36v_low</pre>
;; State determine	rocedure trys to determine which of 5 possible different regions of of Charge that a battery is operating in. _soc_36v PROC FAR USH R0 USH R0		; If soc_36v_low_word > r2_soc_36v_low ; then operating in region 2 JMP cc_C, Region2_36v
Pi Pi Mi Mi	USH R1 USH DPP0 USH DPP1 OV DPP0, #PAG soc_36v_high_word OV DPP1, #PAG r1_soc_36v_high OV R0, DPP0:soc_36v_high_word		; If no Carry must test to see if soc_36v_low_word = r2_soc_36v_low ; If they DON'T equal then soc_36v_low_word < r2_soc_36v_low ; This means region 3 JMP cc_NZ, Region3_36v
M	OV R1, DPP1:rl_soc_36v_high MP R1, R0 ; This subtracts soc_36v_high_word from r1_soc_36v_high so then test		; Getting here means that the soc_36v_high_word = r2_soc_36v_high ; This point is defined to be in Region 2 JMP Region2_36v
;	If there is a carry then soc_36v_high_word was larger than r1_soc_36v_high so a carry was generated soc_36v_high_word > r1_soc_36v_high => Very Dangerous Over Charge => Region 1 MP cc_C, Region1_36v	Test_Reg test fla	gion_3_36v: MOV R1, DPP1:r3_soc_36v_high NOP CMP R1, R0 ; This subtracts soc_36v_high_word from r3_soc_36v_high so then ags
; ; J	If no Carry must test to see if soc_36v_high_word = r1_soc_36v_high If they DON'T equal then soc_36v_high_word < r1_soc_36v_high This means Test for Different Region MP cc_NZ, Test_Region_2_36v		; If there is a carry then soc_36v_high_word was larger ; than r3_soc_36v_high so a carry was generated ; soc_36v_high_word > r3_soc_36v_high => Ideal Operation => Region 3 JMP cc_C, Region3_36v
; M M	Since soc_36v_high_word = r1_soc_36v_high must now test lower word Inorder to determine if battery is in region 1 or region 2 IOV R0, DPP0:soc_36v_low_word IOV R1, DPP1:r1_soc_36v_low IMP R1, R0 ; This subtracts soc_36v_low_word from r1_soc_36v_low		; If no Carry must test to see if soc_36v_high_word = r3_soc_36v_high ; If they DON'T equal then soc_36v_high_word < r3_soc_36v_high ; This means Test for Different Region JMP cc_NZ, Test_Region_4_36v
; J	<pre>If soc_36v_low_word > r1_soc_36v_low then operating in region 1 MP cc_C, Region1_36v If no Carry must test to see if soc_36v_low_word = r1_soc_36v_low</pre>		<pre>; Since soc_36v_high_word = r3_soc_36v_high must now test lower word ; Inorder to determine if battery is in region 2 or region 3 MOV R0, DPP0:soc_36v_low_word MOV R1, DPP1:r3_soc_36v_low CMP R1, R0 ; This subtracts soc_36v_low_word from r3_soc_36v_low</pre>
;	If they DON'T equal then soc_36v_low_word < r1_soc_36v_low This means region 2 MP cc_NZ, Region2_36v		<pre>; If soc_36v_low_word > r3_soc_36v_low ; then operating in region 2 JMP cc_C, Region2_36v</pre>
; J	Getting here means that the soc_36v_high_word = r1_soc_36v_high This point is defined to be in Region 1 IMP Region1_36v		<pre>; If no Carry must test to see if soc_36v_low_word = r3_soc_36v_low ; If they DON'T equal then soc_36v_low_word < r3_soc_36v_low ; This means region 4 IMD ac NZ = Pacient 36u</pre>
M	lon_2_36v: MOV R1, DPP1:r2_soc_36v_high MOP CMP R1, R0 ; This subtracts soc_36v_high_word from r2_soc_36v_high so then test		<pre>JMP cc_NZ, Region4_36v ; Getting here means that the soc_36v_high_word = r3_soc_36v_high ; This point is defined to be in Region 3</pre>
flags .	If there is a carry then soc 36y high word was larger	Test_Re	JMP Region3_36v gion_4_36v:

99/06/14 05:10:20

ema.asm



; Put that number into memory MOV R1. DPP1:r4 soc 36v high MOV DPP0:soc_region_36v, R0 JMP exit soc 36v NOP CMP R1, R0 ; This subtracts soc_36v_high_word from r4_soc_36v_high so then test exit soc 36v: flags POP DPP1 ; If there is a carry then soc_36v_high_word was larger POP DPPO POP R1 ; than r4_soc_36v_high so a carry was generated ; soc_36v_high_word > r4_soc_36v_high => Moderate Undercharge => Region 4 POP RO DET JMP cc C, Region4 36v determine_soc_36v ENDP ; If no Carry must test to see if soc_36v_high_word = r4_soc_36v_high determine soc 12v PROC FAR ; If they DON'T equal then soc_36v_high_word < r4_soc_36v_high PUSH RO ; This means Test for Different Region PUSH R1 JMP cc_NZ, Test_Region_5_36v PUSH DPP0 ; Since soc_36v_high_word = r4_soc_36v_high must now test lower word PUSH DPP1 MOV DPPO, #PAG soc_12v_high_word ; Inorder to determine if battery is in region 2 or region 3 MOV DPP1, #PAG r1_soc_12v_high MOV R0. DPP0:soc 36v low word MOV R0, DPP0:soc_12v_high_word MOV R1, DPP1:r4 soc 36v low ; This subtracts soc_36v_low_word from r4_soc_36v_low MOV R1, DPP1:r1 soc 12v high CMP R1, R0 CMP R1, R0 ; This subtracts soc_12v_high_word from r1_soc_12v_high so then test flags ; If soc_36v_low_word > r4_soc_36v_low ; then operating in region 2 ; If there is a carry then soc_12v_high_word was larger JMP cc_C, Region4_36v ; than r1_soc_12v_high so a carry was generated ; soc_12v_high_word > r1_soc_12v_high => Very Dangerous Over Charge => Regio ; If no Carry must test to see if soc_36v_low_word = r4_soc_36v_low ; If they DON'T equal then soc_36v_low_word < r4_soc_36v_low n 1 JMP cc_C, Region1_12v ; This means region 2 JMP cc NZ, Region5_36v ; If no Carry must test to see if soc_12v_high_word = r1_soc_12v_high ; If they DON'T equal then soc_12v_high_word < r1_soc_12v_high ; Getting here means that the soc_36v_high_word = r4_soc_36v_high ; This means Test for Different Region ; This point is defined to be in Region 2 JMP cc_NZ, Test_Region_2_12v JMP Region4_36v ; Since soc_12v_high_word = r1_soc_12v_high must now test lower word Test_Region_5_36v: ; Inorder to determine if battery is in region 1 or region 2 JMP Region5 36v MOV R0, DPP0:soc_12v_low_word MOV R1, DPP1:r1_soc_12v_low ; This subtracts soc_12v_low_word from r1_soc_12v_low CMP R1, R0 Region1_36v: MOV R0, #01h ; Move the region number into R0 ; If soc_12v_low_word > r1_soc_12v_low ; then operating in region 1 MOV DPP0:soc_region_36v, R0 ; Put that number into memory JMP cc_C, Region1_12v JMP exit_soc_36v ; If no Carry must test to see if soc_12v_low_word = r1_soc_12v_low ; If they DON'T equal then soc_12v_low_word < r1_soc_12v_low Region2_36v: ; This means region 2 MOV R0, #02h ; Move the region number into R0 JMP cc_NZ, Region2_12v MOV DPP0:soc_region_36v, R0 ; Put that number into memory JMP exit_soc_36v ; Getting here means that the soc_12v_high_word = r1_soc_12v_high ; This point is defined to be in Region 1 JMP Region1_12v Region3_36v: MOV R0, #03h ; Move the region number into R0 Test Region 2 12v: MOV DPP0:soc_region_36v, R0 ; Put that number into memory MOV R1, DPP1:r2_soc_12v_high JMP exit_soc_36v NOP CMP R1, R0 ; This subtracts soc_12v_high_word from r2_soc_12v_high so then test flags Region4_36v: MOV R0, #04h ; Move the region number into R0 ; If there is a carry then soc_12v_high_word was larger ; Put that number into memory MOV DPP0:soc_region_36v, R0 ; than r2_soc_12v_high so a carry was generated JMP exit soc 36v ; soc_12v_high_word > r2_soc_12v_high => Very Dangerous Over Charge => Regio n 1 JMP cc C, Region2_12v Region5_36v: MOV R0, #05h ; Move the region number into R0

99/06/14 05:10:20

ema.asm



: If there is a carry then soc_12v_high_word was larger ; If no Carry must test to see if soc_12v_high_word = r2_soc_12v_high ; than r4_soc_12v_high so a carry was generated ; If they DON'T equal then soc_12v_high_word < r2_soc_12v_high ; soc_12v_high_word > r4_soc_12v_high => Moderate Undercharge => Region 4 ; This means Test for Different Region JMP cc C, Region4_12v JMP cc NZ, Test_Region_3_12v : If no Carry must test to see if soc_12v_high_word = r4_soc_12v_high ; Since soc_12v_high_word = r2_soc_12v_high must now test lower word ; If they DON'T equal then soc_12v_high_word < r4_soc_12v_high ; Inorder to determine if battery is in region 2 or region 3 ; This means Test for Different Region MOV R0, DPP0:soc_12v_low_word JMP cc NZ, Test_Region_5_12v MOV R1, DPP1:r2_soc_12v_low ; This subtracts soc_12v_low_word from r2_soc_12v_low CMP R1, R0 ; Since soc_12v_high_word = r4_soc_12v_high must now test lower word ; Inorder to determine if battery is in region 2 or region 3 ; If soc_12v_low_word > r2_soc_12v_low MOV R0, DPP0:soc_12v_low_word ; then operating in region 2 MOV R1, DPP1:r4_soc_12v_low JMP cc_C, Region2_12v CMP R1, R0 ; This subtracts soc_12v_low_word from r4_soc_12v_low ; If no Carry must test to see if soc_12v_low_word = r2_soc_12v_low ; If soc_12v_low_word > r4_soc_12v_low ; If they DON'T equal then soc_12v_low_word < r2_soc_12v_low ; then operating in region 2 ; This means region 3 JMP cc_C, Region4_12v JMP cc_NZ, Region3_12v ; Getting here means that the soc_12v_high_word = r2_soc_12v_high ; If no Carry must test to see if soc_12v_low_word = r4_soc_12v_low ; If they DON'T equal then soc_12v_low_word < r4_soc_12v_low ; This point is defined to be in Region 2 ; This means region 2 JMP Region2 12v JMP cc_NZ, Region5_12v Test_Region_3_12v: ; Getting here means that the soc_12v_high_word = r4_soc_12v_high MOV R1, DPP1:r3_soc_12v_high : This point is defined to be in Region 2 NOP CMP R1, R0 ; This subtracts soc_12v_high_word from r3_soc_12v_high so then test JMP Region4 12v flags Test_Region_5_12v: ; If there is a carry then soc_12v_high_word was larger JMP Region5 12v ; than r3_soc_12v_high so a carry was generated ; soc_12v_high_word > r3_soc_12v_high => Ideal Operation => Region 3 Region1_12v: JMP cc_C, Region3_12v MOV R0, #01h ; Move the region number into R0 MOV DPP0:soc_region_12v, R0 ; Put that number into memory JMP exit_soc_12v ; If no Carry must test to see if soc_12v_high_word = r3_soc_12v_high ; If they DON'T equal then soc_12v_high_word < r3_soc_12v_high Region2_12v: ; This means Test for Different Region MOV R0, #02h ; Move the region number into R0 JMP cc_NZ, Test_Region_4_12v MOV DPP0:soc_region_12v, R0 ; Put that number into memory ; Since soc_12v_high_word = r3_soc_12v_high must now test lower word JMP exit_soc_12v ; Inorder to determine if battery is in region 2 or region 3 Region3_12v: MOV R0, DPP0:soc 12v_low_word MOV R0, #03h ; Move the region number into R0 MOV R1, DPP1:r3 soc 12v low MOV DPP0:soc_region_12v, R0 ; Put that number into memory ; This subtracts soc_12v_low_word from r3_soc_12v_low CMP R1, R0 JMP exit_soc_12v ; If soc_12v_low_word > r3_soc_12v_low Region4_12v: ; then operating in region 2 MOV R0, #04h ; Move the region number into R0 JMP cc_C, Region2_12v MOV DPP0:soc_region_12v, R0 ; Put that number into memory ; If no Carry must test to see if soc_12v_low_word = r3_soc_12v_low JMP exit_soc_12v ; If they DON'T equal then soc_12v_low_word < r3_soc_12v_low ; This means region 4 Region5_12v: MOV R0, #05h ; Move the region number into R0 JMP cc NZ, Region4_12v MOV DPP0:soc_region_12v, R0 ; Put that number into memory : Getting here means that the soc_12v_high_word = r3_soc_12v_high JMP exit_soc_12v ; This point is defined to be in Region 3 exit_soc_12v: JMP Region3_12v POP DPP1 POP DPP0 Test_Region_4_12v: POP R1 MOV R1, DPP1:r4_soc_12v_high POP RO NOP CMP R1, R0 ; This subtracts soc_12v_high_word from r4_soc_12v_high so then test RET determine_soc_12v ENDP flags determine_soc_region ENDS

99/06/14 05:10:20

END





LOCATE main.lno (GENERAL) IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (040000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO main.out







DP8



;** ;** Register	definition	ns for the	e SAB C167
			names and BIT names
.** This file	e can be s	upplied to	o rm166 and a166 (STDNAMES control)
;*********	*******	* * * * * * * * * *	***********
TRUE	DEFB	OFF20h.C	
NODE142	DEFB	0FF20h.1	1, RW
e1 ee5	0000	OPPOOL	
C1CSR	DEFA DEFA	0EF00h 0EF02h	
INTID C1BTR		0EF0211 0EF04h	
CIGMS		0EF06h	
CIUGML		0EF08h	
C1LGML		0EF0Ah	
C1UMLM	DEFA	0EF0Ch	
C1LMLM	DEFA	0EF0Eh	
MCR_M1	DEFA	0EF10h 0EF20h	
MCR_M2	DEFA	0EF20h	
MCR_M3		0EF30h	
MCR_M4	DEFA	0EF40h	
MCR_M5		0EF50h	
MCR_M6		0EF60h	
MCR_M7 MCR_M8	DEFA	0EF70h 0EF80h	
MCR_M8	DEFA	0EF90h	
MCR_MA		0EFA0h	
MCR_MB		0EFB0h	
MCR_MC	DEFA	0EFC0h	
MCR_MD	DEFA	0EFD0h	
MCR_ME	DEFA	0EFE0h 0EFF0h	
MCR_MF	DEFA	0EFF0h	
MCD_M1		0EF16h	
MCD_M2		0EF26h	
MCD_M3		0EF36h	
MCD_M4		0EF46h	
MCD_M5		0EF56h 0EF66h	
MCD_M6 MCD_M7		0EF76h	
MCD_M8	DEFA		
MCD_M9	DEFA		
MCD_MA	DEFA		
MCD_MB		0EFB6h	
MCD_MC	DEFA	0EFC6h	
MCD_MD		0EFD6h	
MCD_ME	DEFA	0EFE6h	
DATA_M1	DEFA	0EF18h	
DATA_M2	DEFA		
DATA_M3	DEFA		OFF49b
DATA_M41		DEFA DEFA	0EF48h 0EF4Ah
DATA_M42 DATA_M5	DEFA		UET 4AII
DATA_M5 DATA_M6	DEFA	0EF68h	
DATA_M7	DEFA	0EF78h	
DATA_M81		DEFA	0EF88h
DATA_M82		DEFA	0EF8Ah
DATA_M9	DEFA	0EF98h	
DATA_MA	DEFA	0EFA8h	
DATA_MB	DEFA	0EFB8h	
DATA_MC	DEFA	0EFC8h	
DATA_MD	DEFA	0EFD8h	
DATA_ME	DEFA	0EFE8h	

DF0	DEFR	OFTDOIL
P8	DEFR	0FFD4h
DP7	DEFR	0FFD2h
P7	DEFR	0FFD0h
DP6		OFFCEh
	DEFR	
P6	DEFR	OFFCCh
DP4	DEFR	OFFCAh
P4	DEFR	0FFC8h
DP3	DEFR	0FFC6h
P3	DEFR	0FFC4h
DP2	DEFR	0FFC2h
P2	DEFR	0FFC0h
SSCCON	DEFR	0FFB2h
SOCON	DEFR	0FFB0h
WDTCON	DEFR	OFFAEh
TFR	DEFR	OFFACh
P5	DEFR	0FFA2h
ADCON	DEFR	0FFA0h
TIIC	DEFR	0FF9Eh
TOIC	DEFR	0FF9Ch
ADEIC	DEFR	0FF9Ah
ADCIC	DEFR	0FF98h
CC15IC	DEFR	0FF96h
CC14IC	DEFR	0FF94h
CC13IC	DEFR	0FF92h
CC12IC	DEFR	0FF90h
CC11IC	DEFR	0FF8Eh
CC10IC	DEFR	0FF8Ch
CC9IC	DEFR	0FF8Ah
CC8IC	DEFR	0FF88h
CC7IC	DEFR	0FF86h
CC6IC	DEFR	0FF84h
CC5IC	DEFR	0FF82h
CC4IC	DEFR	0FF80h
CC3IC	DEFR	0FF7Eh
CC2IC	DEFR	0FF7Ch
CC1IC	DEFR	0FF7Ah
CCOIC	DEFR	0FF78h
SSCEIC	DEFR	0FF76h
SSCRIC	DEFR	0FF74h
SSCTIC	DEFR	0FF72h
SOEIC	DEFR	0FF70h
SORIC	DEFR	0FF6Eh
SOTIC	DEFR	0FF6Ch
CRIC	DEFR	0FF6Ah
TGIC	DEFR	0FF68h
T5IC	DEFR	0FF66h
T4IC	DEFR	0FF64h
T3IC	DEFR	0FF62h
T2IC	DEFR	0FF60h
CCM3	DEFR	0FF58h
CCM2	DEFR	0FF56h
CCM1	DEFR	0FF54h
CCMO	DEFR	0FF52h
TOICON	DEFR	0FF50h
T6CON	DEFR	0FF48h
TSCON	DEFR	0FF46h
T4CON	DEFR	0FF44h
T3CON	DEFR	0FF42h
T2CON	DEFR	0FF40h
PWMCON1	DEFR	0FF32h
PWMCON0	DEFR	0FF30h
CCM7	DEFR	0FF28h
CCM6	DEFR	0FF26h
00110		

0FFD6h

DEFR

99/05/06 15:05:48		
CCM5	DEFR	0FF24h
CCM4	DEFR	0FF22h
T78CON	DEFR	0FF20h
P1H	DEFR	0FF06h
P1L	DEFR	0FF04h
POH	DEFR	0FF02h 0FF00h
POL PECC7	DEFR	OFECEh
PECC6	DEFR	OFECCh
PECC5	DEFR	OFECAh
PECC4	DEFR	0FEC8h
PECC3	DEFR	0FEC6h
PECC2	DEFR	0FEC4h
PECC1	DEFR	0FEC2h
PECCO	DEFR	0FEC0h 0FCE0h
SRCP0 DSTP0	DEFA DEFA	0FCE2h
SRCP1	DEFA	0FCE4h
DSTP1	DEFA	0FCE6h
SRCP2	DEFA	0FCE8h
DSTP2	DEFA	OFCEAh
SRCP3	DEFA	OFCECh
DSTP3	DEFA	OFCEEh
SRCP4	DEFA	0FCF0h 0FCF2h
DSTP4 SRCP5	DEFA DEFA	0FCF2h 0FCF4h
DSTP5	DEFA	0FCF6h
SRCP6	DEFA	0FCF8h
DSTP6	DEFA	OFCFAh
SRCP7	DEFA	OFCFCh
DSTP7	DEFA	OFCFEh
SOBG	DEFR	0FEB4h
SORBUF	DEFR DEFR	OFEB2h, r OFEB0h, w
SOTBUF WDT	DEFR	OFEBOh, w OFEAEh, r
ADDAT	DEFR	0FEA0h
CC15	DEFR	0FE9Eh
CC14	DEFR	0FE9Ch
CC13	DEFR	0FE9Ah
CC12	DEFR	0FE98h
CC11	DEFR DEFR	0FE96h 0FE94h
CC10 CC9	DEFR	0FE92h
CC8	DEFR	0FE90h
CC7	DEFR	0FE8Eh
CC6	DEFR	0FE8Ch
CC5	DEFR	0FE8Ah
CC4	DEFR	0FE88h
CC3	DEFR	0FE86h
CC2	DEFR DEFR	0FE84h 0FE82h
CC1 CC0	DEFR	0FE80h
CC31	DEFR	0FE7Eh
CC30	DEFR	0FE7Ch
CC29	DEFR	0FE7Ah
CC28	DEFR	0FE78h
CC27	DEFR	0FE76h
CC26	DEFR	0FE74h 0FE72h
CC25 CC24	DEFR DEFR	0FE72h 0FE70h
CC24 CC23	DEFR	0FE6Eh
CC22	DEFR	0FE6Ch
CC21	DEFR	0FE6Ah
CC20	DEFR	0FE68h
CC19	DEFR	0FE66h

reg167b.def		
CC1 CC1 CC1 T1R T0R T1 T0 CAP T6 T5 T4	7 DE 6 DE EL DE	FR 0FE62h FR 0FE60h FR 0FE56h FR 0FE52h FR 0FE52h FR 0FE50h FR 0FE4Ah FR 0FE4Ah FR 0FE48h FR 0FE48h
T3 T2 PW3 PW2 PW1 PW0 ; E	DE: DE:	FR 0FE40h FR 0FE36h FR 0FE34h FR 0FE32h FR 0FE30h
ODE ODE ODE ODE PIC ODE EXI S07 XP3 XP2 XP1 XP2 XP1 XP2 XP1 XP2 XP1 XP2 XP1 XP2 CC2 CC2 CC2 CC2 CC2 CC2 CC2 CC2 CC2 C	8 DE 7 DE 6 DE 7 DE 6 DE 2 DE CON DE PIC DE IC DE SIC DE SIC DE 9IC DE 9IC DE 9IC DE H DE H DE H DE ER DE RB DE	FR OF1D6h FR OF1C2h FR OF1C2h FR OF1C2h FR OF1C2h FR OF1C2h FR OF1C2h FR OF12Ch FR OF12Ch FR OF19Ch FR OF19Ch FR OF19Ch FR OF19Ch FR OF19Ch FR OF17Eh FR OF17Ch FR OF17Ah FR OF17Ah FR OF17Ch FR OF17Ah FR OF17Ch FR OF17Ah FR OF17Ch FR
T8F T7F T8 T7 PP3	REL DE DE DE	FR 0F054h FR 0F052h FR 0F050h

15:05:48		
PP2	DEFR	0F03Ch
PP1	DEFR	0F03Ah
PP0	DEFR	0F038h
PT3	DEFR	0F036h
PT2	DEFR	0F034h
PT1	DEFR	0F032h
PTO	DEFR	0F030h
; Bit names		20.0
CCOIO	DEFB	P2.0
CC1IO	DEFB	P2.1 P2.2
CC210 CC310	DEFB DEFB	P2.2 P2.3
CC410	DEFB	P2.4
CC510	DEFB	P2.5
CC6IO	DEFB	P2.6
CC7I0	DEFB	P2.7
CC810	DEFB	P2.8
CC910	DEFB	P2.9
CC10IO	DEFB	P2.10
CC11IO	DEFB	P2.11
CC12I0	DEFB	P2.12
CC13I0	DEFB	P2.13
CC14I0	DEFB	P2.14
CC15I0	DEFB	P2.15
EXOIN	LIT	'CCOIO'
EX1IN	LIT	'CC1IO' 'CC2IO'
EX2IN	LIT LIT	'CC3IO'
EX3IN	LII	
TOIN	DEFB	P3.0
TGOUT	DEFB	P3.1
CAPIN	DEFB	P3.2 P3.3
T3OUT	DEFB DEFB	P3.3 P3.4
T3EUD T2IN	DEFB	P3.4
TJIN	DEFB	P3.6
T4IN	DEFB	P3.5
SSDI	DEFB	P3.8
SSDO	DEFB	P3.9
TXD0	DEFB	P3.10
RXD0	DEFB	P3.11
SSCLK	DEFB	P3.13
CLKOUT	DEFB	P3.15
A16	DEFB	P4.0
A17	DEFB DEFB	P4.1 P4.2
A18 A19	DEFB	P4.2 P4.3
A20	DEFB	P4.4
A21	DEFB	P4.5
A22	DEFB	P4.6
A23	DEFB	P4.7
ANO	DEFB	P5.0
AN1	DEFB	P5.1
AN2	DEFB	P5.2
AN3	DEFB	P5.3
AN4	DEFB DEFB	P5.4 P5.5
AN5 AN6	DEFB	P5.5 P5.6
AN5 AN7	DEFB	P5.0 P5.7
AN8	DEFB	P5.8
AN9	DEFB	P5.9
AN10	DEFB	P5.10

99/05/06

def		
AN11 AN12 AN13 AN14 AN15 T6EUD T5EUD T6IN T5IN T4EUD T2EUD	DEFB DEFB DEFB DEFB LIT LIT LIT LIT LIT	P5.11 P5.12 P5.13 P5.14 P5.15 'AN10' 'AN11' 'AN12' 'AN13' 'AN14' 'AN15'
POUT0 POUT1 POUT2 POUT3 CC28I0 CC29I0 CC29I0 CC30I0 CC31I0	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P7.0 P7.1 P7.2 P7.3 P7.4 P7.5 P7.6 P7.7
CC16I0 CC17I0 CC18I0 CC19I0 CC20I0 CC21I0 CC22I0 CC23I0	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P8.0 P8.1 P8.2 P8.3 P8.4 P8.5 P8.6 P8.7
TOM TOR T1M T1R T7M T7R T8M T8R	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	T01CON.3 T01CON.6 T01CON.11 T01CON.14 T78CON.3 T78CON.6 T78CON.11 T78CON.14
ACC0 ACC1 ACC2 ACC3	DEFB DEFB DEFB DEFB	CCM0.3 CCM0.7 CCM0.11 CCM0.15
ACC4 ACC5 ACC6 ACC7	DEFB DEFB DEFB DEFB	CCM1.3 CCM1.7 CCM1.11 CCM1.15
ACC8 ACC9 ACC10 ACC11	DEFB DEFB DEFB DEFB	CCM2.3 CCM2.7 CCM2.11 CCM2.15
ACC12 ACC13 ACC14 ACC15	DEFB DEFB DEFB DEFB	CCM3.3 CCM3.7 CCM3.11 CCM3.15
ACC16 ACC17 ACC18 ACC19	DEFB DEFB DEFB DEFB	CCM4.3 CCM4.7 CCM4.11 CCM4.15

3

9	9/	0	5/	0	6	
1	5:	0	5:	4	8	

ACC20	DEFB	CCM5.3	SSCTIE	DEFB	SSCTIC.6
ACC21	DEFB	CCM5.7	SSCTIR	DEFB	SSCTIC.7
ACC22	DEFB	CCM5.11	SSCRIE	DEFB	SSCRIC.6
ACC23	DEFB	CCM5.15	SSCRIR	DEFB	SSCRIC.7
needo			SSCEIE	DEFB	SSCEIC.6
ACC24	DEFB	ССМ6.3	SSCEIR	DEFB	SSCEIC.7
ACC25	DEFB	ССМ6.7	SSCTE	LIT	'SSCTEN'
ACC26	DEFB	CCM6.11	SSCRE	LIT	'SSCREN'
ACC27	DEFB	ССМ6.15	SSCPE	LIT	'SSCPEN'
			SSCBE	LIT	'SSCBEN'
ACC28	DEFB	CCM7.3			
ACC29	DEFB	CCM7.7			
ACC30	DEFB	CCM7.11	CCOIE	DEFB	CCOIC.6
ACC31	DEFB	CCM7.15	CCOIR	DEFB	CCOIC.7
			CC1IE	DEFB	CC1IC.6
T2R	DEFB	T2CON.6	CC1IR	DEFB	CC1IC.7
T2UD	DEFB	T2CON.7	CC2IE	DEFB	CC2IC.6
T2UDE	DEFB	T2CON.8	CC2IR	DEFB	CC2IC.7
			CC3IE	DEFB	CC3IC.6
T3R	DEFB	T3CON.6	CC3IR	DEFB	CC3IC.7
T3UD	DEFB	T3CON.7	CC4IE CC4IR	DEFB DEFB	CC4IC.6 CC4IC.7
TJUDE	DEFB	T3CON.8	CC5IE	DEFB	CC5IC.6
T3OE	DEFB	T3CON.9		DEFB	CC5IC.7
TJOTL	DEFB	T3CON.10	CC5IR CC6IE	DEFB	CCGIC.6
m 4 D	DDDD		CCGIR	DEFB	CCGIC.7
T4R	DEFB	T4CON.6	CC7IE	DEFB	CC7IC.6
T4UD	DEFB	T4CON.7	CC7IE CC7IR	DEFB	CC7IC.7
T4UDE	DEFB	T4CON.8	CC8IE	DEFB	CC8IC.6
	DDDD	T5CON.6	CC8IR	DEFB	CC8IC.7
T5R	DEFB		CC9IE	DEFB	CC9IC.6
T5UD	DEFB	T5CON.7	CC9IR	DEFB	CC9IC.7
T5UDE	DEFB	T5CON.8	CC10IE	DEFB	CC10IC.6
T5CLR	DEFB	T5CON.14	CC10IR	DEFB	CC10IC.7
T5SC	DEFB	T5CON.15	CC11IE	DEFB	CC11IC.6
mcD	DEFB	T6CON.6	CC11IR	DEFB	CC11IC.7
T6R		16CON. 7	CC12IE	DEFB	CC12IC.6
TGUD	DEFB	T6CON.8	CC12IR	DEFB	CC12IC.7
TOUDE	DEFB DEFB	T6CON.9	CC13IE	DEFB	CC13IC.6
TGOE	DEFB	T6CON.10	CC13IR	DEFB	CC13IC.7
T60TL T6SR	DEFB	T6CON.15	CC14IE	DEFB	CC14IC.6
TOSK	DEFD	10001.15	CC14IR	DEFB	CC14IC.7
T2IE	DEFB	T2IC.6	CC15IE	DEFB	CC15IC.6
T2IR	DEFB	T2IC.7	CC15IR	DEFB	CC15IC.7
TJIE	DEFB	T3IC.6	CC16IE	DEFB	CC16IC.6
T3IR	DEFB	T3IC.7	CC16IR	DEFB	CC16IC.7
T4IE	DEFB	T4IC.6	CC17IE	DEFB	CC17IC.6
T4IR	DEFB	T4IC.7	CC17IR	DEFB	CC17IC.7
TSIE	DEFB	T5IC.6	CC18IE	DEFB	CC18IC.6
T5IR	DEFB	T5IC.7	CC18IR	DEFB	CC18IC.7
TGIE	DEFB	T6IC.6	CC19IE	DEFB	CC19IC.6
T6IR	DEFB	T6IC.7	CC19IR	DEFB	CC19IC.7
			CC20IE	DEFB	CC20IC.6
CRIE	DEFB	CRIC.6	CC20IR	DEFB	CC20IC.7
CRIR	DEFB	CRIC.7	CC21IE	DEFB	CC21IC.6
			CC21IR	DEFB	CC21IC.7
SOTIE	DEFB	SOTIC.6	CC22IE	DEFB	CC22IC.6
SOTIR	DEFB	SOTIC.7	CC22IR	DEFB	CC22IC.7
SORIE	DEFB	SORIC.6	CC23IE	DEFB	CC23IC.6
SORIR	DEFB	SORIC.7	CC23IR	DEFB	CC23IC.7
SOEIE	DEFB	SOEIC.6	CC24IE	DEFB	CC24IC.6
SOEIR	DEFB	SOEIC.7	CC24IR	DEFB	CC24IC.7
SOTBIE	DEFB	SOTBIC.6	CC25IE	DEFB	CC25IC.6
SOTBIR	DEFB	SOTBIC.7	CC25IR	DEFB	CC25IC.7
			CC26IE	DEFB	CC26IC.6

99/05/06 15:05:48		
CC26IR	DEFB	CC26IC.7
CC27IE	DEFB	CC27IC.6
CC27IR	DEFB	CC27IC.7
CC28IE	DEFB	CC28IC.6
CC28IR	DEFB	CC28IC.7
CC29IE	DEFB	CC29IC.6
CC29IR	DEFB	CC29IC.7
	DEFB	CC30IC.6
CC30IE		CC301C.7
CC30IR	DEFB	
CC31IE	DEFB	CC31IC.6
CC31IR	DEFB	CC31IC.7
ADCIE	DEFB	ADCIC.6
ADCIR	DEFB	ADCIC.7
ADEIE	DEFB	ADEIC.6
ADEIR	DEFB	ADEIC.7
TOIE	DEFB	TOIC.6
TOIR	DEFB	TOIC.7
T1IE	DEFB	TIIC.6
T1IR	DEFB	T1IC.7
T7IE	DEFB	T7IC.6
T7IR	DEFB	T7IC.7
T8IE	DEFB	T8IC.6
T8IR	DEFB	T8IC.7
ADST	DEFB	ADCON.7
ADBSY	DEFB	ADCON.8
ADWR	DEFB	ADCON.9
ADCIN	DEFB	ADCON.10
ADCRQ	DEFB	ADCON.11
ILLBUS	DEFB	TFR.0
ILLINA	DEFB	TFR.1
	DEFB	TFR.2
ILLOPA		TFR.3
PRTFLT	DEFB	
UNDOPC	DEFB	TFR.7
STKUF	DEFB	TFR.13
STKOF	DEFB	TFR.14
NMI	DEFB	TFR.15
WDTIN	DEFB	WDTCON.0
WDTR	DEFB	WDTCON.1
SOSTP	DEFB	SOCON.3
SOREN	DEFB	SOCON.4
SOPEN	DEFB	SOCON.5
SOFEN	DEFB	SOCON.6
SOOEN	DEFB	SOCON.7
SOPE	DEFB	SOCON.8
SOFE	DEFB	SOCON.9
SOOE	DEFB	SOCON.10
SOODD	DEFB	SOCON.12
SOBRS	DEFB	SOCON.13
SOLB	DEFB	SOCON.14
SOR	DEFB	SOCON.15
SSCHB	DEFB	SSCCON.4
SSCPH	DEFB	SSCCON.5
SSCPO	DEFB	SSCCON.6
SSCTEN	DEFB	SSCCON.8
SSCREN	DEFB	SSCCON.9
	DEFB	SSCCON.10
SSCPEN	DEFB	SSCCON.10
SSCBEN		
SSCBSY	DEFB	SSCCON.12

reg167b.def							
	SSCMS	DEFB	SSCCON.14				
	SSCEN	DEFB	SSCCON.15				
	PTR0	DEFB	PWMCON0.0				
	PTR1	DEFB	PWMCON0.1				
	PTR2	DEFB	PWMCON0.2				
	PTR3	DEFB	PWMCON0.3				
	PTIO	DEFB	PWMCON0.4				
	PTI1	DEFB	PWMCON0.5				
	PTI2	DEFB	PWMCON0.6				
	PTI3	DEFB	PWMCON0.7				
	PIEO	DEFB	PWMCON0.8				
	PIE1	DEFB	PWMCON0.9				
	PIE2	DEFB	PWMCON0.10				
	PIE3	DEFB	PWMCON0.11				
	PIRO	DEFB	PWMCON0.12				
	PIR1	DEFB	PWMCON0.13				
	PIR2	DEFB	PWMCON0.14				
	PIR3	DEFB	PWMCON0.15				
	PENO	DEFB	PWMCON1.0				
	PEN1	DEFB	PWMCON1.1				
	PEN2	DEFB	PWMCON1.2				
	PEN3	DEFB	PWMCON1.3				
	PM0	DEFB	PWMCON1.4				
	PM1	DEFB	PWMCON1.5				
	PM2	DEFB	PWMCON1.6				
	PM3	DEFB	PWMCON1.7				
	PB01	DEFB	PWMCON1.12				
	PS2	DEFB	PWMCON1.14				
	PS3	DEFB	PWMCON1.15				
	PWMIE	DEFB	PWMIC.6				
	PWMIR	DEFB	PWMIC.7				
	XP3IE	DEFB	XP3IC.6				
	XP3IR	DEFB	XP3IC.7				
	XP2IE	DEFB	XP2IC.6				
	XP2IR	DEFB	XP2IC.7				
	XP1IE	DEFB	XP1IC.6				
	XP1IR	DEFB	XP1IC.7				
	XPOIE	DEFB	XPOIC.6				
	XPOIR	DEFB	XPOIC.7				



al66 main.asm al66 canmod.asm al66 canmo.asm al66 canint.asm ll66 LINK main.obj canmod.obj canmo.obj canint.obj TO main.lno ll66 @linker.lnv ihex166 -il6 main.out -o main.hex



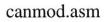


main.asm



RET ; return \$SEGMENTED main ENDP SEXTEND mainseg ENDS \$EXTSFR : CAN USE ALL internal RAM for Stack SEXTSSK ; codesegment that contains reset int pointer startupsec SECTION CODE SEXTMEM ; reset interrupt number is zero at Oh \$NOMOD166 sysreset PROC TASK INTNO=0H ; forces next instruction to be located at Oh ORG 000H \$STDNAMES(reg167b.def) ; installs a pointer to the startup routine JMP start \$SYMBOLS RETI ; return from interrupt sysreset ENDP NAME main startupsec ENDS ; define a common register area of 16 register RBANK1 COMREG R0-R15 END : default stack size of 256 Words SSKDEF 4 ASSUME DPP3:SYSTEM ; Can function EXTERN canin: FAR mainseg SECTION CODE main PROC FAR start: DISWDT : disable the watchdog timer : Globally Enable Interrupts both global BSET IEN ;; Initialize the External Memory BUS MOV SYSCON, #0E084h MOV ADDRSEL1, #0404h MOV BUSCONO, #004AFh MOV BUSCON1, #004AFh EINIT ; end initialization ;; End of external memory bus initialization ;; Initialize the Data Page pointers for this section ; make DPP3 point to system MOV DPP3, #03h ;; End of Data Page Pointer Initialization ;; Make sure Port 2 is in Open Drain mode MOV ODP2, ONES ;; Make the direction of Port 2 to output MOV DP2, ONES ;; Make sure all of the ports are off MOV P2, ONES BCLR P2.8 ;; Initialize The Stack ;; The Stack pointers are all word pointers so even though the ;; highest byte in the stack is located at #OFBFFh the highest ;; byte that the stack pointers can point to is #OFBFEh MOV STKUN, #OFBFEh; Set Stack Underflow Pointer MOV STKOV, #0F800h; Set STack Overflow Pointer MOV SP, #0FBFEh ; Set the Stack Pointer ;; End of Stack Initialization ;; Initialize CAN Bus ; Call the CAN initialization function CALL canin ;; End of CAN Bus Initialization meto: NOP ; just loop here waiting NOP JMP meto

99/05/09 11:00:32





	1000		cannou.asin			
SEGMENTED				RET		
\$EXTEND			canin	ENDP		
ŞEXTSFR						
ŞEXTMEM			setall	PROC FAR	; This Procedure sets all of the Mess	s objs invalid
\$NOMOD166					it counts up to 15 and initializes all	. of the message
\$STDNAMES(reg	167b.def)			;; objects along the wa	ay.	
\$SYMBOLS				PUSH R2		
				PUSH R4		
NAME canmod				PUSH R5		
DDAW1 COMDE	C D0 D15	; define a common register area of 16 registers		AND R5,ZEROS OR R5, #01h	; Set counter to 1 for first MO	
RBANK1 COMRE GLOBAL canin		; The function must be declared Global at the		AND R2, ZEROS	AL MARK INCOMENTATION	
GLOBAL Callin		; beginning of the module		OR R2, #0EF10h	; Set pointer to MO1	
		,		AND R4, ZEROS		
EXTERN canmo	cfg:FAR	; configures specific Message objects		OR R4, #5555h	; Set R4 to make MObs invalid	
ASSUME DPP3:S	YSTEM		nextreg	:MOV [R2],R4	; make all message objects invalid	
6	2000	and a sector that contains react int pointor		ADD R2,#10h CMPI1 R5,#0Fh		
canfunc SECT	ION CODE	; codesegment that contains reset int pointer		JMPA CC_NZ, nextreg	;	
canin PROC	EVD			POP R5	1	
PUSH				POP R4		
PUSH				POP R2		
				RET		
	et all of the CAN		setall	ENDP		
		et control register to zero	-			
	1, #0043h	; Set IE and INIT bits	canfund END	ENDS		
OR C1	.CSR,R1 ; set	control register to R1's value	END			
AND C		et Bit timing register to zero				
		; set for 125k operation				
		Bit timing register parameters				
		et Global Mask short register to zero				
	1, #0FFFFh	; EOFF is what DAVE initialize				
OR C1	.GMS, R1 ; set	GMS				
AND C	THOM TEDOS	et Upper global mask long to zero				
	1, #0FFFFh	to opper grobar mask rong to bere				
	UGML, R1					
25 G10 - 546						
MOV R	R1, #0F8FFh					
	CILGML, ZEROS	where the loss of the second to a				
OR C1	LGML, R1	; lower global mask				
	ATTAL A BEDOC					
1.1	CIUMLM, ZEROS LUMLM, R1	; upper mask of last register				
	CILMLM, ZEROS	, apper main or rate regreeor				
	LMLM, R1	; lower mask of last register				
CALL	setall	; sets all of the CAN registers to off				
1014 AD7 (17 102)	1111111111111111 1 111	a si				
CALL	canmocfg	; Configures specific Message Objects				
	atun CAN interrunt	and Initialize CAN module				
EXTR #4						
AND X	KPOIC, ZEROS ; co	onfigure CAN interrupt control Register				
	R0,ZEROS					
),#0073h	; enable interrupt, level is 10 group is 2				
		figure CAN interrupt Control Register				
AND F	R1, ZEROS	the if I aloon the ODI access to the DMD				
	L, #00041h ; cras	shes if I clear the CPU access to the BTR				
		initialize CAN interrupt				
POP F POP F						
POP P						
			1			



canmo.asm



;; This message object transmits the present state of the DC/DC converter SSEGMENTED SEXTEND ; start of Message Object 3 MOV R2, #MCR_M3 SEXTSFR AND R1, ZEROS SEXTMEM ; RECEIVE INTERRUPT NOT enabled OR R1, #5595h \$NOMOD166 ; set MO2's Control register MOV [R2], R1 \$STDNAMES(reg167b.def) ADD R2,#2h ; point to Upper Arbitration register \$SYMBOLS AND R3, ZEROS ; set R6 to zero OR R3, #000Fh ; The number is the Message ID for Message Object 2 NAME canmo MOV [R2], R3 ; message id = 000F ; declare bank of 16 global registers RBANK1 COMREG R0-R15 ; Point to the Lower Arbitration Register ADD R2, #2h GLOBAL canmocfg ; standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS : put 00038h into first data byte and set to transmi OR R1, #0038h can_module SECTION CODE t ; Databyte(0) = 0 and Set to transmit and 3 bytes MOV MCD_M3,R1 ASSUME DPP3:SYSTEM of data ; Fill the Data of the MO with Zeros MOV DATA_M3, ZEROS canmocfg PROC FAR PUSH R1 PUSH R2 PUSH R3 ;; Now set specific CAN control Registers POP R3 POP R2 ;; initialize message object 1 POP R1 ;; initializing this object to be invalid does or removing the code until ;; the comment "Setup CAN interrupt and Initialize " does RET canmocfg ENDP ;; nothing to prevent the occurrance of the interrupt for the CAN system can module ENDS ; start of Message Object 1 MOV R2, #MCR M1 END AND R1, ZEROS ; Generate a Receive Interrupt if this message object ac OR R1, #5599h tivates ; set MO1's Control register MOV [R2], R1 ; point to Upper Arbitration register ADD R2, #2h AND R3, ZEROS ; set R3 to ; message id for message object 1 OR R3, #000Eh ; message id = #000Eh MOV [R2], R3 ; Point to the Lower Arbitration Register ADD R2, #2h MOV [R2], ZEROS ; standard Message object so lowerarb = 0h AND R1, ZEROS ; put 00h into first data byte and set to receive OR R1, #0030h ; Databyte(0) = 0 and Set to receive and 3 bytes of data MOV MCD_M1, R1 ; fill the Data of the MO with Zeros MOV DATA_M1, ZEROS ;; Initialize Message Object 2 ;; This message object receives information about turning the DC/DC converter on and off ;; For the purpose of the thesis the DC/DC was just left on all the time. MOV R2, #MCR_M2 ; start of Message Object 2 AND R1, ZEROS ; RECEIVE INTERRUPT NOT enabled OR R1, #5599h MOV [R2], R1 ; set MO3's Control register ADD R2,#2h ; point to Upper Arbitration register ; set R6 to zero AND R3, ZEROS ; The number is the Message ID for Message Object 3 OR R3, #0021h ; message id = 00021h MOV [R2], R3 ; Point to the Lower Arbitration Register ADD R2, #2h : standard Message object so lowerarb = 0h MOV [R2], ZEROS AND R1, ZEROS ; put 00030h into first data byte and set to receive OR R1, #0030h ; Databyte(0) = 0 and Set to transmit and 3 bytes of d MOV MCD M2, R1 ata ; Fill the Data of the MO with Zeros MOV DATA M2, ZEROS ;; Initialize Message Object 3

99/05/14 11:39:24

\$SEGMENTED \$EXTEND \$EXTSFR \$EXTMEM \$NOMOD166 \$STDNAMES(reg167b.def) \$SYMBOLS NAME canint RBANK1 COMREG R0-R15 ; declare bank of 16 global registers ASSUME DPP3:SYSTEM can_interrupts SECTION CODE can_receive_interrupt PROC TASK INTNO=040h ORG 0100h CALL can_receive_interrupt_handler RETI can_receive_interrupt ENDP can_receive_interrupt_handler PROC FAR PUSH RO PUSH R1 PUSH R2 PUSH R3 PUSH R4 MOVB RLO, INTID ; Read the CAN interrupt ID buffer CMPB RLO, #03h ; See if the interrupt came from M01 JMP cc_Z, message_one_interrupt; if interrupt from M01 handle MOV R1, #05555h MOV R2, #05599h MOV MCR_M2, R1 MOV RO, DATA_M2 MOV R3, R0 ; Put the Data in R3 for future use MOV MCR_M2, R2 CMP R0, #01h JMP cc_NZ, turn_off_converter ;; This is where the converter is turned on MOV R4, P2 BSET R4.8 MOV P2, R4 JMP exit function turn_off_converter: CMP R0, #0800h JMP cc_NZ, exit_function MOV R4, P2 BCLR R4.8 MOV P2, R4 JMP exit_function message_one_interrupt: ;; Message Object one deals with the state of the DC/DC converter MOV R1, #05555h MOV R2, #05599h MOV MCR_M1, R1 MOV RO, DATA_M1 MOV MCR_M1, R2 ;; Now setup M3 so it can respond to queries about ;; the state of the converter

MOV R2, MCR_M3 MOV MCR_M3, R1 MOV DATA_M3, R0 MOV MCR_M3, R2 MOV R3, DATA_M3 MOV R4, P2 MOVB RL4, RL3 MOV P2, R4 ; This is where the DC/DC converter is actually set. exit_function: POP R4 POP R3 POP R2 POP R1 POP RO RET can_receive_interrupt_handler ENDP can_interrupts ENDS END

canint.asm



LOCATE main.lno (GENERAL) IRAMSIZE (2048) RESERVE MEMORY(0F200h TO 0F5FFh) MEMORY(ROM (0000h to 0EFFFh), RAM (040000h to 4EFFFh), IRAM(0F000h)) CLASSES('RAM' (040000h to 04FFFFh)) SYMBOLS LISTSYMBOLS TO main.out



99/05/08 23:03:14

reg167b.def

P8

; * * * * * * * * * * * * *	******	* * * * * * * * * * * *	* * * * * * * * *	* * * * * * * * * *	* * * * * * * * * *	* * * * * * * * * *
;** @(#)reg167		1.10 12/1				
; * *						
;** Register d						
;** This file						2.23
;** This file	can be	supplied to	rm166 and	a166 (STI	DNAMES con	trol)
; *************				********	* * * * * * * * * * *	*******
TRUE	DEFB	0FF20h.0,				
NODE142	DEFB	0FF20h.1,	RW			
C1CSR	DEFA	0EF00h				
INTID	DEFA	0EF02h				
CIBTR	DEFA	0EF04h				
CIGMS	DEFA	0EF06h				
C1UGML	DEFA	0EF08h				
C1LGML	DEFA	0EF0Ah				
C1UMLM	DEFA	0EF0Ch				
C1LMLM	DEFA	0EF0Eh				
MCR_M1	DEFA	0EF10h				
MCR_M2	DEFA	0EF20h				
MCR_M3	DEFA	0EF30h				
MCR_M4 MCR_M5	DEFA DEFA	0EF40h 0EF50h				
MCR_M5 MCR_M6	DEFA	0EF60h				
MCR_M7	DEFA	0EF70h				
MCR_M8	DEFA	0EF80h				
MCR_M9	DEFA	0EF90h				
MCR_MA	DEFA	0EFA0h				
MCR_MB	DEFA	0EFB0h				
MCR_MC	DEFA	0EFC0h				
MCR_MD	DEFA	0EFD0h				
MCR_ME	DEFA	OEFEOh				
MCR_MF MCD_M1	DEFA DEFA	0EFF0h 0EF16h				
MCD_M1 MCD_M2	DEFA	0EF16H				
MCD_M2 MCD_M3	DEFA	0EF36h				
MCD_M4	DEFA	0EF46h				
MCD_M5	DEFA	0EF56h				
MCD_M6	DEFA	0EF66h				
MCD_M7	DEFA	0EF76h				
MCD_M8	DEFA	0EF86h				
MCD_M9	DEFA	0EF96h				
MCD_MA	DEFA	0EFA6h				
MCD_MB	DEFA	0EFB6h				
MCD_MC	DEFA	0EFC6h 0EFD6h				
MCD_MD MCD_ME	DEFA DEFA	0EFE6h				
DATA_M1	DEFA	0EF18h				
DATA_M1 DATA_M2	DEFA	0EF28h				
DATA_M3	DEFA	0EF38h				
DATA_M4	DEFA	0EF48h				
DATA_M5	DEFA	0EF58h				
DATA_M6	DEFA	0EF68h				
DATA_M7	DEFA	0EF78h				
DATA_M8	DEFA	0EF88h				
DATA_M9	DEFA	0EF98h				
DATA_MA	DEFA	0EFA8h				
DATA_MB	DEFA	0EFB8h				
DATA_MC	DEFA	0EFC8h 0EFD8h				
DATA_MD DATA_ME	DEFA DEFA	0EFE8h				
DATA_ME	DEFA	VEI EOII				
DP8	DEFR	0FFD6h				

Po	DEFR	OFFDan
DP7	DEFR	0FFD2h
P7	DEFR	OFFDOh
DP6	DEFR	OFFCEh
P6	DEFR	OFFCCh
DP4	DEFR	OFFCAh
P4	DEFR	0FFC8h
DP3	DEFR	0FFC6h
P3	DEFR	0FFC4h
DP2	DEFR	0FFC2h
P2	DEFR	OFFCOh
SSCCON	DEFR	0FFB2h
SOCON	DEFR	OFFBOh
WDTCON	DEFR	OFFAEh
TFR	DEFR	OFFACh
P5	DEFR	0FFA2h
ADCON	DEFR	OFFAOh
TIIC	DEFR	0FF9Eh
TOIC	DEFR	0FF9Ch
ADEIC	DEFR	0FF9Ah
ADCIC	DEFR	0FF98h
CC15IC	DEFR	0FF96h
CC14IC	DEFR	0FF94h
CC13IC	DEFR	0FF92h
CC12IC	DEFR	0FF90h
		0FF8Eh
CC11IC	DEFR	
CC10IC	DEFR	0FF8Ch
CC9IC	DEFR	0FF8Ah
	DEFR	0FF88h
CC7IC	DEFR	0FF86h
CC6IC	DEFR	0FF84h
CC5IC	DEFR	0FF82h
CC4IC	DEFR	0FF80h
CC3IC	DEFR	0FF7Eh
		0FF7Ch
CC2IC	DEFR	
CC1IC	DEFR	0FF7Ah
CCOIC	DEFR	0FF78h
SSCEIC	DEFR	0FF76h
SSCRIC	DEFR	0FF74h
SSCTIC	DEFR	0FF72h
SOEIC	DEFR	0FF70h
SORIC	DEFR	0FF6Eh
SOTIC	DEFR	0FF6Ch
CRIC	DEFR	0FF6Ah
TGIC	DEFR	0FF68h
T5IC	DEFR	0FF66h
T4IC	DEFR	0FF64h
T3IC	DEFR	0FF62h
T2IC	DEFR	0FF60h
CCM3	DEFR	0FF58h
CCM2	DEFR	0FF56h
CCM1	DEFR	0FF54h
	DEFR	0FF52h
CCM0		
T01CON	DEFR	0FF50h
T6CON	DEFR	0FF48h
T5CON	DEFR	0FF46h
T4CON	DEFR	0FF44h
T3CON	DEFR	0FF42h
T2CON	DEFR	0FF40h
PWMCON1	DEFR	0FF32h
PWMCON0	DEFR	0FF30h
CCM7	DEFR	0FF28h
CCM6	DEFR	0FF26h
CCM5	DEFR	0FF24h
CCM4	DEFR	0FF22h

DEFR

0FFD4h

99/05/08 23:03:14		
T78CON	DEFR	0FF20h
P1H	DEFR	0FF06h
P1L	DEFR	0FF04h
POH	DEFR	0FF02h
POL	DEFR	0FF00h
PECC7	DEFR	OFECEh
PECC6	DEFR	OFECCh
PECC5	DEFR	0FECAh 0FEC8h
PECC4 PECC3	DEFR	0FEC6h
PECC2	DEFR	0FEC4h
PECC1	DEFR	0FEC2h
PECC0	DEFR	0FEC0h
SRCP0	DEFA	0FCE0h
DSTP0	DEFA	0FCE2h
SRCP1	DEFA	0FCE4h
DSTP1	DEFA	0FCE6h
SRCP2	DEFA	0FCE8h 0FCEAh
DSTP2 SRCP3	DEFA DEFA	OFCEAN
DSTP3	DEFA	OFCEEh
SRCP4	DEFA	0FCF0h
DSTP4	DEFA	0FCF2h
SRCP5	DEFA	0FCF4h
DSTP5	DEFA	0FCF6h
SRCP6	DEFA	0FCF8h
DSTP6	DEFA	OFCFAh
SRCP7	DEFA	OFCFCh
DSTP7	DEFA	0FCFEh 0FEB4h
SOBG SORBUF	DEFR DEFR	OFEB4h
SOTBUF	DEFR	OFEBOh,
WDT	DEFR	OFEAEh,
ADDAT	DEFR	OFEA0h
CC15	DEFR	0FE9Eh
CC14	DEFR	0FE9Ch
CC13	DEFR	0FE9Ah
CC12	DEFR	0FE98h
CC11	DEFR DEFR	0FE96h 0FE94h
CC10 CC9	DEFR	0FE94H
CC8	DEFR	0FE90h
CC7	DEFR	0FE8Eh
CC6	DEFR	0FE8Ch
CC5	DEFR	0FE8Ah
CC4	DEFR	0FE88h
CC3	DEFR	0FE86h
CC2	DEFR	0FE84h
CC1	DEFR	OFE82h
CC0	DEFR DEFR	0FE80h 0FE7Eh
CC31 CC30	DEFR	0FE7Eh
CC29	DEFR	0FE7Ah
CC28	DEFR	0FE78h
CC27	DEFR	0FE76h
CC26	DEFR	0FE74h
CC25	DEFR	0FE72h
CC24	DEFR	OFE70h
CC23	DEFR	OFE6Eh
CC22 CC21	DEFR DEFR	0FE6Ch 0FE6Ah
CC20	DEFR	0FE68h
CC19	DEFR	0FE66h
CC18	DEFR	0FE64h
CC17	DEFR	0FE62h

r w

r

CC16 DEFR 0FE60h 0FE56h T1REL DEFR 0FE54h TOREL DEFR DEFR 0FE52h т1 0FE50h т0 DEFR DEFR 0FE4Ah CAPREL DEFR 0FE48h т6 т5 DEFR 0FE46h т4 DEFR 0FE44h 0FE42h Т3 DEFR т2 DEFR 0FE40h PW3 DEFR 0FE36h PW2 DEFR 0FE34h 0FE32h PW1 DEFR DEFR 0FE30h PW0 ; Extended sfr area ODP8 DEFR 0F1D6h ODP7 DEFR 0F1D2h ODP6 DEFR 0F1CEh ODP3 DEFR 0F1C6h PICON DEFR 0F1C4h ODP2 DEFR 0F1C2h EXICON DEFR 0F1C0h SOTBIC DEFR 0F19Ch XP3IC DEFR 0F19Eh DEFR 0F196h XP2IC XP1IC DEFR 0F18Eh XPOIC DEFR 0F186h PWMIC DEFR OF17Eh T8IC DEFR 0F17Ch T7IC DEFR 0F17Ah CC31IC DEFR 0F194h CC30IC DEFR 0F18Ch 0F184h CC29IC DEFR CC28IC DEFR 0F178h CC27IC DEFR 0F176h CC26IC DEFR 0F174h CC25IC DEFR 0F172h CC24IC DEFR 0F170h CC23IC DEFR 0F16Eh CC22IC DEFR 0F16Ch CC21IC DEFR 0F16Ah CC20IC DEFR 0F168h CC19IC DEFR 0F166h CC18IC DEFR 0F164h CC17IC DEFR 0F162h CC16IC DEFR 0F160h RPOH DEFR 0F108h DEFR 0F106h DP1H DP1L DEFR 0F104h DEFR 0F102h DPOH DPOL DEFR 0F100h SSCBR DEFR 0F0B4h SSCRB DEFR 0F0B2h SSCTB DEFR OFOBOh ADDAT2 DEFR OFOAOh T8REL DEFR 0F056h T7REL 0F054h DEFR DEFR 0F052h т8 т7 0F050h DEFR PP3 DEFR 0F03Eh PP2 DEFR 0F03Ch 0F03Ah PP1 DEFR



reg167b.def

99	/05	/08
23	:03	:14

PPO					
		0F038h	AN13	DEFB	P5.13
	DEFR		AN14	DEFB	P5.14
PT3	DEFR	0F036h			
PT2	DEFR	0F034h	AN15	DEFB	P5.15
PT1	DEFR	0F032h	T6EUD	LIT	'AN10'
PTO	DEFR	0F030h	T5EUD	LIT	'AN11'
FIO	DELIX		TGIN	LIT	'AN12'
					'AN13'
; Bit names			T5IN	LIT	
CC0IO	DEFB	P2.0	T4EUD	LIT	'AN14'
CC1IO	DEFB	P2.1	T2EUD	LIT	'AN15'
CC2IO	DEFB	P2.2			
		P2.3	POUTO	DEFB	P7.0
CC310	DEFB				
CC4IO	DEFB	P2.4	POUT1	DEFB	P7.1
CC510	DEFB	P2.5	POUT2	DEFB	P7.2
CC6IO	DEFB	P2.6	POUT3	DEFB	P7.3
CC710	DEFB	P2.7	CC28I0	DEFB	P7.4
			CC2910	DEFB	P7.5
CC810	DEFB	P2.8			
CC9IO	DEFB	P2.9	CC30I0	DEFB	P7.6
CC10I0	DEFB	P2.10	CC31IO	DEFB	P7.7
CC11IO	DEFB	P2.11			
CC12I0	DEFB	P2.12	CC16I0	DEFB	P8.0
		P2.13	CC17I0	DEFB	P8.1
CC13I0	DEFB			DEFB	P8.2
CC14IO	DEFB	P2.14	CC18I0		
CC15I0	DEFB	P2.15	CC19I0	DEFB	P8.3
EXOIN	LIT	, CC010,	CC20I0	DEFB	P8.4
EX1IN	LIT	· CC110 ′	CC21I0	DEFB	P8.5
	LIT	'CC2I0'	CC22I0	DEFB	P8.6
EX2IN				DEFB	P8.7
EX3IN	LIT	'CC3IO'	CC23I0	DEFB	P0./
TOIN	DEFB	P3.0			
TGOUT	DEFB	P3.1	TOM	DEFB	T01CON.3
			TOR	DEFB	T01CON.6
CAPIN	DEFB	P3.2		DEFB	T01CON.11
TJOUT	DEFB	P3.3	T1M		
T3EUD	DEFB	P3.4	T1R	DEFB	T01CON.14
T2IN	DEFB	P3.7	T7M	DEFB	T78CON.3
T3IN	DEFB	P3.6	T7R	DEFB	T78CON.6
		P3.5	T8M	DEFB	T78CON.11
T4IN	DEFB				T78CON.14
SSDI	DEFB	P3.8	T8R	DEFB	1/00011.14
	DEFB	P3.9			
SSDO					CCM0.3
		P3.10	ACC0	DEFB	00110.5
TXD0	DEFB	P3.10 P3.11			CCM0.7
TXD0 RXD0	DEFB DEFB	P3.11	ACC1	DEFB	CCM0.7
TXD0 RXD0 SSCLK	DEFB DEFB DEFB	P3.11 P3.13	ACC1 ACC2	DEFB DEFB	CCM0.7 CCM0.11
TXD0 RXD0	DEFB DEFB	P3.11	ACC1	DEFB	CCM0.7
TXD0 RXD0 SSCLK CLKOUT	DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15	ACC1 ACC2 ACC3	DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15
TXD0 RXD0 SSCLK	DEFB DEFB DEFB	P3.11 P3.13	ACC1 ACC2 ACC3 ACC4	DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3
TXDO RXDO SSCLK CLKOUT A16	DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15	ACC1 ACC2 ACC3	DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15
TXDO RXDO SSCLK CLKOUT A16 A17	DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1	ACC1 ACC2 ACC3 ACC4	DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18	DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6	DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3	ACC1 ACC2 ACC3 ACC4 ACC5	DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7	DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7	DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.5 P4.6	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC9 ACC10	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4 AN5	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14 ACC15	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC13 ACC14 ACC15 ACC16	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4 AN5 AN6	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14 ACC15	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4 AN5 AN6 AN7	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC13 ACC14 ACC15 ACC16	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN1 AN2 AN3 AN4 AN5 AN6 AN7 AN8	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7 P5.8	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14 ACC15 ACC16 ACC17 ACC18	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM3.15 CCM4.3 CCM4.7 CCM4.11
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4 AN5 AN6 AN7 AN8 AN9	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7 P5.8 P5.9	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC12 ACC13 ACC14 ACC15 ACC16 ACC17	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4 AN5 AN6 AN7 AN8 AN9 AN10	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.5 P5.6 P5.7 P5.8 P5.9 P5.10	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC13 ACC14 ACC15 ACC16 ACC17 ACC18 ACC18 ACC19	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7 CCM4.11 CCM4.15
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4 AN5 AN6 AN7 AN8 AN9	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.6 P5.7 P5.8 P5.9 P5.10 P5.11	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC13 ACC13 ACC14 ACC15 ACC16 ACC17 ACC18 ACC19 ACC19 ACC20	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7 CCM4.11 CCM4.15 CCM5.3
TXD0 RXD0 SSCLK CLKOUT A16 A17 A18 A19 A20 A21 A22 A23 AN0 AN1 AN2 AN3 AN4 AN5 AN6 AN7 AN8 AN9 AN10	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	P3.11 P3.13 P3.15 P4.0 P4.1 P4.2 P4.3 P4.4 P4.5 P4.6 P4.7 P5.0 P5.1 P5.2 P5.3 P5.4 P5.5 P5.5 P5.6 P5.7 P5.8 P5.9 P5.10	ACC1 ACC2 ACC3 ACC4 ACC5 ACC6 ACC7 ACC8 ACC9 ACC10 ACC11 ACC12 ACC13 ACC14 ACC15 ACC16 ACC17 ACC18 ACC18 ACC19	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CCM0.7 CCM0.11 CCM0.15 CCM1.3 CCM1.7 CCM1.11 CCM1.15 CCM2.3 CCM2.7 CCM2.11 CCM2.15 CCM3.3 CCM3.7 CCM3.11 CCM3.15 CCM4.3 CCM4.7 CCM4.11 CCM4.15

99/05/08
23:03:14

4

Contraction of the second						
				CCCRIP	DDDD	CCODIC 6
ACC22	DEFB	CCM5.11		SSCRIE	DEFB	SSCRIC.6
ACC23	DEFB	CCM5.15		SSCRIR	DEFB	SSCRIC.7
				SSCEIE	DEFB	SSCEIC.6
ACC24	DEFB	CCM6.3	1	SSCEIR	DEFB	SSCEIC.7
ACC25	DEFB	CCM6.7		SSCTE	LIT	'SSCTEN'
				SSCRE	LIT	'SSCREN'
ACC26	DEFB	CCM6.11				'SSCPEN'
ACC27	DEFB	CCM6.15		SSCPE	LIT	
				SSCBE	LIT	'SSCBEN'
ACC28	DEFB	CCM7.3				
ACC29	DEFB	CCM7.7				
ACC30	DEFB	CCM7.11		CCOIE	DEFB	CCOIC.6
				CCOIR	DEFB	CCOIC.7
ACC31	DEFB	CCM7.15				
				CC1IE	DEFB	CC1IC.6
T2R	DEFB	T2CON.6		CC1IR	DEFB	CC1IC.7
T2UD	DEFB	T2CON.7		CC2IE	DEFB	CC2IC.6
T2UDE	DEFB	T2CON.8		CC2IR	DEFB	CC2IC.7
IZODE	DELD	12000.0		CC3IE	DEFB	CC3IC.6
		72 00N (CC3IR	DEFB	CC3IC.7
T3R	DEFB	T3CON.6				
TJUD	DEFB	T3CON.7		CC4IE	DEFB	CC4IC.6
TJUDE	DEFB	T3CON.8		CC4IR	DEFB	CC4IC.7
TJOE	DEFB	T3CON.9		CC5IE	DEFB	CC5IC.6
TJOTL	DEFB	T3CON.10		CC5IR	DEFB	CC5IC.7
15011	DEFD	1500N.10		CC6IE	DEFB	CC6IC.6
		m10011 (CC6IR	DEFB	CC6IC.7
T4R	DEFB	T4CON.6				
T4UD	DEFB	T4CON.7		CC7IE	DEFB	CC7IC.6
T4UDE	DEFB	T4CON.8		CC7IR	DEFB	CC7IC.7
				CC8IE	DEFB	CC8IC.6
T5R	DEFB	T5CON.6		CC8IR	DEFB	CC8IC.7
				CC91E	DEFB	CC9IC.6
T5UD	DEFB	T5CON.7				
T5UDE	DEFB	T5CON.8		CC9IR	DEFB	CC9IC.7
T5CLR	DEFB	T5CON.14		CC10IE	DEFB	CC10IC.6
T5SC	DEFB	T5CON.15		CC10IR	DEFB	CC10IC.7
				CC11IE	DEFB	CC11IC.6
mcp	DEFB	T6CON.6		CC11IR	DEFB	CC11IC.7
T6R				CC12IE	DEFB	CC12IC.6
TGUD	DEFB	T6CON.7	81			
T6UDE	DEFB	T6CON.8		CC12IR	DEFB	CC12IC.7
TGOE	DEFB	T6CON.9		CC13IE	DEFB	CC13IC.6
TGOTL	DEFB	T6CON.10		CC13IR	DEFB	CC13IC.7
T6SR	DEFB	T6CON.15		CC14IE	DEFB	CC14IC.6
103K	DBID	10001115		CC14IR	DEFB	CC14IC.7
	0000	m210 (CC15IE	DEFB	CC15IC.6
T2IE	DEFB	T2IC.6			DEFB	CC15IC.7
T2IR	DEFB	T2IC.7		CC15IR		
TJIE	DEFB	T3IC.6		CC16IE	DEFB	CC16IC.6
T3IR	DEFB	T3IC.7		CC16IR	DEFB	CC16IC.7
T4IE	DEFB	T4IC.6		CC17IE	DEFB	CC17IC.6
T4IR	DEFB	T4IC.7		CC17IR	DEFB	CC17IC.7
				CC18IE	DEFB	CC18IC.6
T5IE	DEFB	T5IC.6			DEFB	CC18IC.7
T5IR	DEFB	T5IC.7		CC18IR		
T6IE	DEFB	T6IC.6		CC19IE	DEFB	CC19IC.6
T6IR	DEFB	T6IC.7		CC19IR	DEFB	CC19IC.7
				CC20IE	DEFB	CC20IC.6
CRIE	DEFB	CRIC.6		CC20IR	DEFB	CC20IC.7
				CC21IE	DEFB	CC21IC.6
CRIR	DEFB	CRIC.7				CC21IC.7
				CC21IR	DEFB	
SOTIE	DEFB	SOTIC.6		CC22IE	DEFB	CC22IC.6
SOTIR	DEFB	SOTIC.7		CC22IR	DEFB	CC22IC.7
SORIE	DEFB	SORIC.6		CC23IE	DEFB	CC23IC.6
		SORIC.7		CC23IR	DEFB	CC23IC.7
SORIR	DEFB			CC24IE	DEFB	CC24IC.6
SOEIE	DEFB	SOEIC.6				
SOEIR	DEFB	SOEIC.7		CC24IR	DEFB	CC24IC.7
SOTBIE	DEFB	SOTBIC.6		CC25IE	DEFB	CC25IC.6
SOTBIR	DEFB	SOTBIC.7		CC25IR	DEFB	CC25IC.7
JV10111				CC26IE	DEFB	CC26IC.6
COORTE	DEED	CCOMIC 6		CC26IR	DEFB	CC26IC.7
SSCTIE	DEFB	SSCTIC.6		CC27IE	DEFB	CC27IC.6
SSCTIR	DEFB	SSCTIC.7		CC2/IE	DEFB	CC2/1C.0

23:03:14	1	
CC27IR CC28IE CC28IR CC29IE CC29IR CC30IE CC30IR CC31IE CC31IR	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	CC27IC.7 CC28IC.6 CC28IC.7 CC29IC.6 CC29IC.7 CC30IC.6 CC30IC.7 CC30IC.7 CC31IC.6 CC31IC.7
ADCIE ADCIR ADEIE ADEIR	DEFB DEFB DEFB DEFB	ADCIC.6 ADCIC.7 ADEIC.6 ADEIC.7
T0IE T0IR T1IE T1IR T7IE T7IR T8IE T8IR	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	TOIC.6 TOIC.7 TIIC.6 TIIC.7 T7IC.6 T7IC.7 T8IC.6 T8IC.7
ADST ADBSY ADWR ADCIN ADCRQ	DEFB DEFB DEFB DEFB	ADCON.7 ADCON.8 ADCON.9 ADCON.10 ADCON.11
ILLBUS ILLINA ILLOPA PRTFLT UNDOPC STKUF STKOF NMI	DEFB DEFB DEFB DEFB DEFB DEFB DEFB	TFR.0 TFR.1 TFR.2 TFR.3 TFR.7 TFR.13 TFR.14 TFR.15
WDTIN WDTR	DEFB DEFB	WDTCON.0 WDTCON.1
SOSTP SOREN SOPEN SOFEN SOPE SOFE SOOE SOODD SOBRS SOLB SOR	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	SOCON.3 SOCON.4 SOCON.5 SOCON.6 SOCON.7 SOCON.8 SOCON.9 SOCON.10 SOCON.12 SOCON.13 SOCON.14 SOCON.15
SSCHB SSCPH SSCPO SSCTEN SSCREN SSCBEN SSCBEN SSCBSY SSCMS SSCEN	DEFB DEFB DEFB DEFB DEFB DEFB DEFB DEFB	SSCCON.4 SSCCON.5 SSCCON.6 SSCCON.8 SSCCON.9 SSCCON.10 SSCCON.11 SSCCON.12 SSCCON.14 SSCCON.15

99/05/08

reg167b.def

PTRO	DEFB	PWMCON0.0
PTR1	DEFB	PWMCON0.1
PTR2	DEFB	PWMCON0.2
PTR3	DEFB	PWMCON0.3
PTIO	DEFB	PWMCON0.4
PTI1	DEFB	PWMCON0.5
PTI2	DEFB	PWMCON0.6
PTI3	DEFB	PWMCON0.7
PIEO	DEFB	PWMCON0.8
PIE1	DEFB	PWMCON0.9
PIE2	DEFB	PWMCON0.10
PIE3	DEFB	PWMCON0.11
PIRO	DEFB	PWMCON0.12
PIR1	DEFB	PWMCON0.13
PIR2	DEFB	PWMCON0.14
PIR3	DEFB	PWMCON0.15
PENO	DEFB	PWMCON1.0
PEN1	DEFB	PWMCON1.1
PEN2	DEFB	PWMCON1.2
PEN3	DEFB	PWMCON1.3
PM0	DEFB	PWMCON1.4
PM1	DEFB	PWMCON1.5
PM2	DEFB	PWMCON1.6
PM3	DEFB	PWMCON1.7
PB01	DEFB	PWMCON1.12
PS2	DEFB	PWMCON1.14
PS3	DEFB	PWMCON1.15
PWMIE	DEFB	PWMIC.6
PWMIR	DEFB	PWMIC.7
XP3IE	DEFB	XP3IC.6
XP3IR	DEFB	XP3IC.7
XP2IE	DEFB	XP2IC.6
XP2IR	DEFB	XP2IC.7
XP1IE	DEFB	XP1IC.6
XP1IR	DEFB	XP1IC.7
XPOIE	DEFB	XPOIC.6
XPOIR	DEFB	XPOIC.7



B.11 Saber to Breadboard Converter Code

On the next page starts the code for the Java Saber to Breadboard Converter tool. The files for the node are as follows.

- 1. SaberConverter.java
- 2. SaberFrame.java
- 3. SaberFrame_AboutBox.java

B.12 Breadboard Loads

On the next page is the file BreadBoardLoads.txt

saberc~1.~ja

99/05/24 10:30:26

```
Saber to Bread Board Converter
//Title:
//Version:
//Copyright: Copyright (c) 1998
              James Geraci
//Author:
              MIT LEES Lab
//Company:
//Description:Saber to Bread Board Converter
package Thesis;
import com.sun.java.swing.UIManager;
import java.awt.*;
import java.io.*;
import java.util.*;
import java.text.*;
import borland.jbcl.util.*;
public class SaberConverter {
  boolean packFrame = false;
  //Construct the application
  public SaberConverter() {
    SaberFrame frame = new SaberFrame();
    //Validate frames that have preset sizes
    //Pack frames that have useful preferred size info, e.g. from their layout
    if (packFrame)
      frame.pack();
    else
      frame.validate();
    //Center the window
    Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
    Dimension frameSize = frame.getSize();
    if (frameSize.height > screenSize.height)
      frameSize.height = screenSize.height;
    if (frameSize.width > screenSize.width)
      frameSize.width = screenSize.width;
    frame.setLocation((screenSize.width - frameSize.width) / 2, (screenSize.height - frameSize.height) / 2);
    frame.setVisible(true);
//Main method
  public static void main(String[] args) {
    try {
    // UIManager.setLookAndFeel(new com.sun.java.swing.plaf.windows.WindowsLookAndFeel());
      //UIManager.setLookAndFeel(new com.sun.java.swing.plaf.motif.MotifLookAndFeel());
      UIManager.setLookAndFeel(new com.sun.java.swing.plaf.metal.MetalLookAndFeel());
```

99/05/24 10:30:26



```
}
    catch (Exception e) {
    }
    new SaberConverter();
 }
}
class AlternatorRPMObject
  {
   public AlternatorRPMObject(TextField WheelDiameter, TextField DiffGearR, TextField EngAltGearR, String s)
      {
        VehicleDrivingSpeed = 0;
        TireDiameter = new Double(WheelDiameter.getText().trim()).doubleValue();
        DifferentialGearRatio = new Double(DiffGearR.getText().trim()).doubleValue();
        TransmissionGearRatio = 0;
        EngineAlternatorGearRatio = new Double(EngAltGearR.getText().trim()).doubleValue();
        AlternatorShaftSpeed = 0;
        TimeOfEvent = 0;
        try
          {
            GenerateAlternatorShaftSpeed(s);
        catch(IOException rt)
          {
            System.exit(1);
          }
      }
  public void GenerateAlternatorShaftSpeed(String s) throws IOException
      {
        String Time;
        String Speed;
        String Gear;
        StringTokenizer token = new StringTokenizer(s, " \t\n\r");
        if(token.hasMoreTokens())
        {
        Time = token.nextToken();
        Speed = token.nextToken();
        Gear = token.nextToken();
        int TimeLength = Time.length();
        int SpeedLength = Speed.length();
        int GearLength = Gear.length();
```



saberc~1.~ja



```
double TimeDataDouble = new Double(Time.substring(0, TimeLength)).doubleValue();
double SpeedDataDouble = new Double(Speed.substring(0, SpeedLength)).doubleValue();
double GearDataDouble = new Double(Gear.substring(0, GearLength)).doubleValue();
VehicleDrivingSpeed = SpeedDataDouble;
int TimeDataInteger = (int) (TimeDataDouble);
int SpeedDataInteger = (int) (SpeedDataDouble);
int GearDataInteger = (int) (GearDataDouble );
if(GearDataInteger == 0 && SpeedDataInteger != -1)
  {
    TransmissionGearRatio = 0 ;
  }
else if(GearDataInteger == 1 && SpeedDataInteger != -1)
  {
    TransmissionGearRatio = 3.071;
  }
else if(GearDataInteger == 2 && SpeedDataInteger != -1)
    TransmissionGearRatio = 1.773;
  }
else if(GearDataInteger == 3 && SpeedDataInteger != -1)
  {
    TransmissionGearRatio = 1.194;
  }
else if(GearDataInteger == 4 && SpeedDataInteger != -1)
  {
    TransmissionGearRatio = 0.868;
  }
else if(GearDataInteger == 5 && SpeedDataInteger != -1)
    TransmissionGearRatio = 0.700;
  }
else
  {
    TransmissionGearRatio = 0;
  }
if (GearDataInteger == 0 && SpeedDataInteger != -1)
  {
    AlternatorShaftSpeed = 600;
else if (SpeedDataInteger == -1)
    AlternatorShaftSpeed = 0;
  }
```





```
else
          {
          AlternatorShaftSpeed = (TransmissionGearRatio*((10.0/36.0)*(60.0)/(TireDiameter*(Math.PI)))*DifferentialGearRatio*V
ehicleDrivingSpeed);
          if((AlternatorShaftSpeed == 0) || (AlternatorShaftSpeed < 600))
              AlternatorShaftSpeed = 600;
            }
          3
        Time = Integer.toString(TimeDataInteger);
        Speed = Integer.toString(SpeedDataInteger);
        Gear = Integer.toString(GearDataInteger);
       AlternatorSpeed = Integer.toString((int) AlternatorShaftSpeed);
       }
      }
      public String getAlternatorShaftSpeed()
      {
        //return VehicleDrivingSpeed;
        return AlternatorSpeed;
      }
        public double getAlternatorShaftSpeed2()
      {
        //return VehicleDrivingSpeed;
        return AlternatorShaftSpeed;
      }
    private String AlternatorSpeed;
    private double VehicleDrivingSpeed;
    private double TireDiameter;
    private double DifferentialGearRatio;
    private double TransmissionGearRatio;
    private double EngineAlternatorGearRatio;
    private double AlternatorShaftSpeed;
    private double TimeOfEvent;
  }
class CANEventObject
  {
    CANEventObject()
      {
      }
```





99/05/24 10:30:26

int indexofdecimalpoint = 0; int indexoflastfrontslash = 0; int indexofopenbrace = 0;int indexofclosebrace = 0; String NameOfCANEvent; String OnandOffTimes; int CANEventVectorSize = 0; int CANCounterSize = 0; int counter = 0;int counter2 = 0;boolean AlreadyExists = false; String AppendFileName; s.trim(); if(!(s.startsWith("#"))) { if(s.startsWith("alter")) indexofdecimalpoint = s.indexOf("."); indexoflastfrontslash = s.lastIndexOf("/"); indexofopenbrace = s.indexOf("["); indexofclosebrace = s.indexOf("]"); NameOfCANEvent = s.substring((indexofdecimalpoint + 1), indexoflastfrontslash); CANCounterSize = CANEventGenerators.size(); while(counter2 < CANCounterSize) { if (NameOfCANEvent.equals(((CANObjectClass) (CANEventGenerators.elementAt(counter2))).returnCANEventName())) { CANEventVectorSize = counter2; counter2 = CANCounterSize + 1; AlreadyExists = true; } counter2++; } if(!AlreadyExists) CANEventGenerators.addElement(new CANObjectClass(NameOfCANEvent)); if(CANEventGenerators.size() != 0) CANEventVectorSize = CANEventGenerators.size() - 1; } if(indexofclosebrace != -1) { ((CANObjectClass) (CANEventGenerators.elementAt(CANEventVectorSize))).setOnandOffTimes(s.substring(indexofopen brace + 1, indexofclosebrace));



```
}
            else
              {
               ((CANObjectClass) (CANEventGenerators.elementAt(CANEventVectorSize))).setOnandOffTimes(s.substring(indexofopen
brace + 1));
            }
            if(s.startsWith("(") || s.startsWith(","))
                if(indexofclosebrace != -1)
                  ((CANObjectClass) (CANEventGenerators.elementAt(CANEventVectorSize))).setOnandOffTimes(s.substring(indexofo
penbrace + 1, indexofclosebrace));
                  }
                else
                  {
                  ((CANObjectClass) (CANEventGenerators.elementAt(CANEventVectorSize))).setOnandOffTimes(s.substring(indexofo
penbrace + 1));
                  }
              }
        }
        else if(s.startsWith("#") && s.endsWith(".scs"))
          {
           StringTokenizer token = new StringTokenizer(s, " \t\n\r");
           int TokenCount = token.countTokens();
           int x = 1;
              while(x < TokenCount)</pre>
                  if(token.hasMoreElements())
                     {
                     token.nextToken();
                     }
                    x++;
          ((CANObjectClass) (CANEventGenerators.elementAt(CANEventVectorSize))).setNameOfExtensionFile(token.nextToken());
         ((CANObjectClass) (CANEventGenerators.elementAt(CANEventVectorSize))).setDoExtensionFilesExist(true);
         3
    }
 }
    public void trimVectors()
      {
       CANEventGenerators.trimToSize();
    public int returnCANEventListSize()
      {
          return CANEventGenerators.size();
    11
```





```
// return BreadBoardCANLoads.size();
 11
       return ValidCANEventGenerators.size();
11
      return ProgrammableLoadPowerDemanded.size();
11
      return FinalCANList.size();
      }
 public String returnProgrammableLoad(int x)
    {
    return ((String) ProgrammableLoadPowerDemanded.elementAt(x));
    }
    public String returnCANString(int x)
      {
      // return ((CANObjectClass)(CANEventGenerators.elementAt(x))).returnOnandOffTimes();
        return ((String) (BreadBoardCANLoads.elementAt(x)));
   11
    return (((CANMessageClass)(FinalCANList.elementAt(x))).returnCANMessage());
        return ((CANObjectClass)(ValidCANEventGenerators.elementAt(x))).returnCANEventName();
   11
      }
public int returnCANEventTime(int x)
  {
    return (((CANMessageClass)(FinalCANList.elementAt(x))).returntime());
  }
public void ReadinBreadBoardCANLoads(String inputfile)
      {
       try{
        BufferedReader filein = new BufferedReader(new FileReader(inputfile.trim()));
        String s;
// Here is where the actual load list is read in.
        while((s = filein.readLine()) != null)
          {
            handleCANString(s);
          }
          filein.close();
        }
       catch(IOException ex)
        {
        System.exit(1);
        }
      }
```



```
private void handleCANString(String x)
    StringTokenizer token = new StringTokenizer(x, " \t\n\r");
   if(token.hasMoreElements())
      String CanName1 = (String) token.nextElement();
      String Message_ID = (String) token.nextElement();
      // Here is where the load is actually added to the list
      BreadBoardCANLoads.addElement(new CANObjectClass(CanName1, Message_ID));
      }
  }
public void ConfirmBreadBoardCompatability()
    1
      while(CANEventGenerators.size() > 0)
        {
      // System.out.println(CANEventGenerators.size() + "\t" + ValidCANEventGenerators.size() + "\t" + NotValidCANEventGener
ators.size());
        boolean istrue = false;
        int BreadBoardLoads = BreadBoardCANLoads.size() - 1;
        int count = 0;
        int holder = 0;
        while(count < BreadBoardLoads)
          {
            if((((CANObjectClass)(BreadBoardCANLoads.elementAt(count))).returnCANEventName()).equals(((CANObjectClass) (CANEv
entGenerators.elementAt(0))).returnCANEventName()))
              {
                istrue = true;
                holder = count;
              }
            count++;
          }
          if(istrue)
            ValidCANEventGenerators.addElement((CANObjectClass) (CANEventGenerators.elementAt(0)));
            ((CANObjectClass) ValidCANEventGenerators.lastElement()).setMessageID(((CANObjectClass)(BreadBoardCANLoads.elemen
tAt(holder))).returnMessageID());
            CANEventGenerators.removeElementAt(0);
             }
            else
              {
                NotValidCANEventGenerators.addElement((CANObjectClass) (CANEventGenerators.elementAt(0)));
                CANEventGenerators.removeElementAt(0);
              }
```





```
}
ValidCANEventGenerators.trimToSize();
```

}

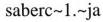
public void GenerateEMValvePowerDemand(TextField HigherVoltage, TextField LowerVoltage, TextField PowerAvailable, Vector Al ternatorSpeedVector)

```
{
    double HigherBusVoltage = new Double(HigherVoltage.getText().trim()).doubleValue();
    double LowerBusVoltage = new Double(LowerVoltage.getText().trim()).doubleValue();
    double ProgrammablePowerAvailable = 1800; //new Double(PowerAvailable.getText().trim()).doubleValue();
    double IdleRPMSpeed = 600; // From Irene Quo's Master Thesis page 85 of motor not alternator
    double HighSpeedRPMSpeed = 2000; // From Irene Quo's Master Thesis page 85
    double MaxCurrent = ProgrammablePowerAvailable / HigherBusVoltage;
    double MinCurrent = MaxCurrent / 5;
    double SizeofAlternatorSpeedVector = AlternatorSpeedVector.size();
    double Slope = (MaxCurrent - MinCurrent)/(HighSpeedRPMSpeed - IdleRPMSpeed);
      System.out.println("This is Computing the EMValve Power Demanded");
 11
 System.out.println("Slope = " + Slope);
    int counter = 0;
     while(counter < SizeofAlternatorSpeedVector)</pre>
        {
         if((Double.valueOf((((AlternatorRPMObject) (AlternatorSpeedVector.elementAt(counter))).getAlternatorShaftSpeed())).
doubleValue()) < IdleRPMSpeed)</pre>
             {
              ProgrammableLoadPowerDemanded.addElement(Double.toString(0));
          else if((Double.valueOf((((AlternatorRPMObject) (AlternatorSpeedVector.elementAt(counter))).getAlternatorShaftSpeed
())).doubleValue()) >= HighSpeedRPMSpeed)
                ProgrammableLoadPowerDemanded.addElement(Double.toString(MaxCurrent));
          else
              int Current = (int) (Slope*((Double.valueOf((((AlternatorRPMObject) (AlternatorSpeedVector.elementAt(counter)))
.getAlternatorShaftSpeed())).doubleValue())) - 6.425);
              if (Current <= MaxCurrent)
                {
                  ProgrammableLoadPowerDemanded.addElement(Double.toString(Current));
                }
              else
                {
                  ProgrammableLoadPowerDemanded.addElement(Double.toString(MaxCurrent));
                }
         counter++;
```

99/05/24 10:30:26



```
}
   // System.out.println("EMValve stuff computed");
   ProgrammableLoadPowerDemanded.trimToSize();
   }
   public void CreateCANMessages()
      int y = 0;
      while(y < ValidCANEventGenerators.size())</pre>
        {
        parseCANString(((CANObjectClass) (ValidCANEventGenerators.elementAt(y))).returnOnandOffTimes(), ((CANObjectClass) Val
idCANEventGenerators.elementAt(y)));
          System.out.println(((CANObjectClass) (ValidCANEventGenerators.elementAt(y))).returnOnandOffTimes());
   11
        y++;
        }
     }
    private void parseCANString(String s, CANObjectClass sClass)
      StringTokenizer token = new StringTokenizer(s, "(");
      // Test to see if Tokens exist
            while(token.hasMoreElements())
              String snext = token.nextToken();
              SemiFinalCANList.addElement(new CANMessageClass(snext, sClass));
               }
      }
     public void removeCANString(int x)
      {
        FinalCANList.removeElementAt(x);
      }
     public void removeT0offMessages()
     {
      while(SemiFinalCANList.size() != 0)
        {
        if((((CANMessageClass) SemiFinalCANList.elementAt(0)).returntime()) == 0)
            if(!(((CANMessageClass) (SemiFinalCANList.elementAt(0))).returnTurnOn()))
              {
                SemiFinalCANList.removeElementAt(0);
              }
        else
```





```
FinalCANList.addElement((CANMessageClass) (SemiFinalCANList.elementAt(0)));
           SemiFinalCANList.removeElementAt(0);
         3
        }
     }
     private String CANString;
     private Vector FinalCANList = new Vector(100, 20);
     private Vector SemiFinalCANList = new Vector(100, 20);
     private Vector BreadBoardCANLoads = new Vector(11);
     private Vector CANEventGenerators = new Vector(100, 20);
     private Vector ValidCANEventGenerators = new Vector(11);
     private Vector NotValidCANEventGenerators = new Vector(11);
     private Vector ProgrammableLoadPowerDemanded = new Vector(30000, 500);
    // private Vector ProgrammableLoadLoads = new Vector(11);
}
class CANMessageClass
  {
   CANMessageClass(String s, CANObjectClass CANObject)
      StringTokenizer token = new StringTokenizer(s, ",");
        if(token.hasMoreElements())
          {
            time = (int) Integer.parseInt(token.nextToken());
          String e = token.nextToken();
        StringTokenizer token2 = new StringTokenizer(e, ")");
// Compute most of the checksum
          byte[] buffer = new byte[4];
                buffer = (CANObject.returnMessageID()).getBytes();
                int idvalue = ConvertFromText(buffer);
                int checksum = 0;
                checksum = 3 + 8 + idvalue; // not done with the checksum just yet
        // Determine if you are turning the switch on or off
          e = (token2.nextToken()).trim();
         if(e.equals("2") || e.equals("3") || e.equals("4"))
              {
                checksum = checksum + 1;
                String HexString = ConvertToHex(checksum);
                CANMessage = "A00308"+ (CANObject.returnMessageID()) + "000001000000000" + HexString + "0A"; // Turn the 42V
olt Heater On
                TurnOn = true;
```

99/05/24 10:30:26



```
}
              else if (e.equals("1"))
              {
                checksum = checksum + 8;
                String HexString = ConvertToHex(checksum);
                CANMessage = "A00308" + (CANObject.returnMessageID()) + "00080000000000" + HexString + "0A"; // Turn the 42
Volt Heater Off
                TurnOn = false;
              }
          }
        }
   private int ConvertFromText(byte[] byter)
      {
        int sum = 0;
        int int0 = (int) byter[0];
        int int2 = (int) byter[2];
        int int3 = (int) byter[3];
        if(int0 >= 48 && int0 <= 57)
         {
            int0 = int0 - 48;
          }
        else if(int0 >= 65 && int0 <= 70)
          {
            int0 = int0 - 55;
          }
        if(int2 >= 48 && int2 <= 57)
          {
            int2 = int2 - 48;
          }
        else if(int2 >= 65 && int2 <= 70)
          {
            int2 = int2 - 55;
          }
        if(int3 >= 48 && int3 <= 57)
          {
            int3 = int3 - 48;
          }
        else if(int3 >= 65 && int3 <= 70)
          {
            int3 = int3 - 55;
          }
        int0 = int0 * 16;
        int2 = int2 * 16;
        sum = int0 + int2 + int3;
```





```
return sum;
  }
private String ConvertToHex(int checksum)
  {
   char[] chararray = new char[4];
   int lowestnibble = checksum & 15;
   int secondnibble = checksum & 240;
   int thirdnibble = checksum & 3840;
   int topnibble = checksum & 61440;
   secondnibble = secondnibble >>> 4;
   thirdnibble = thirdnibble >>> 8;
   topnibble = topnibble >>> 12;
   chararray[0] = FindLetter(topnibble);
   chararray[1] = FindLetter(thirdnibble);
   chararray[2] = FindLetter(secondnibble);
   chararray[3] = FindLetter(lowestnibble);
   String sammy = new String(chararray);
   return sammy;
  }
private char FindLetter(int x)
  {
    char y = '0';
    if(x >= 10)
      {
      y = (char) (x + 55);
      }
    else if(x <= 9)
      {
        y = (char) (x + 48);
      }
    return y;
    }
public boolean returnTurnOn()
  {
    return TurnOn;
  }
public int returntime()
  {
    return time;
  }
```

99/05/24 10:30:26



```
public String returnCANMessage()
      {
        return CANMessage;
      }
    private int time;
    private boolean TurnOn;
    private String CANMessage;
    }
class CANObjectClass
  {
   CANObjectClass(String CName, String M_ID)
      {
        CANEventName = CName;
        Message_ID = M_ID;
      }
    CANObjectClass(String s)
      {
       CANEventName = s;
      }
    public void setOnandOffTimes(String s)
      {
        String s2 = s.trim();
        if(!append)
          {
            OnandOffTimes = s;
            append = true;
          }
        else
          {
            OnandOffTimes = OnandOffTimes + s;
          }
      }
    public void appendOnandOffTimes(String s)
      {
        OnandOffTimes = OnandOffTimes + s;
      }
    public void setNameOfExtensionFile(String s)
      {
        NameOfExtensionFile = s;
```

99/05/24 10:30:26

}



```
}
public String returnNameOfExtensionFile()
  {
    return NameOfExtensionFile;
    }
public String returnOnandOffTimes()
  {
    return OnandOffTimes;
  }
public String returnCANEventName()
  {
    return CANEventName;
  }
public boolean returnDoExtensionFilesExist()
  {
    return DoExtensionFilesExist;
  }
public void setDoExtensionFilesExist(boolean t)
  {
    DoExtensionFilesExist = t;
  }
public void setMessageID(String s)
  {
    Message_ID = s;
  }
public String returnMessageID()
  {
    return Message_ID;
  }
private boolean append = false;
private String CANEventName;
private String OnandOffTimes;
private String NameOfExtensionFile = "No Extension Files";
private String Message_ID;
private boolean DoExtensionFilesExist = false;
private Vector OffTimes = new Vector(20);
```





```
e.printStackTrace();
```



```
2
```

```
}
pack();
}
```

```
private void jbInit() throws Exception {
  this.setTitle("About");
  setResizable(false);
  panel1.setLayout(borderLayout1);
  panel2.setLayout(borderLayout2);
  insetsPanel1.setLayout(flowLayout1);
  insetsPanel1.setBevelInner(BevelPanel.FLAT);
  insetsPanel2.setLayout(flowLayout1);
  insetsPanel2.setMargins(new Insets(10, 10, 10, 10));
  insetsPanel2.setBevelInner(BevelPanel.FLAT);
  gridLayout1.setRows(4);
  gridLavout1.setColumns(1);
  label1.setText(product);
  label2.setText(version);
  label3.setText(copyright);
  label4.setText(comments);
  insetsPanel3.setLayout(gridLayout1);
  insetsPanel3.setMargins(new Insets(10, 60, 10, 10));
  insetsPanel3.setBevelInner(BevelPanel.FLAT);
  button1.setText("OK");
  button1.addActionListener(this);
  imageControl1.setImageName("");
  insetsPanel2.add(imageControl1, null);
  panel2.add(insetsPanel2, BorderLayout.WEST);
  this.add(panel1, null);
  insetsPanel3.add(label1, null);
  insetsPanel3.add(label2, null);
  insetsPanel3.add(label3, null);
  insetsPanel3.add(label4, null);
  panel2.add(insetsPanel3, BorderLayout.CENTER);
  insetsPanel1.add(button1, null);
  panel1.add(insetsPanel1, BorderLayout.SOUTH);
  panel1.add(panel2, BorderLayout.NORTH);
}
protected void processWindowEvent(WindowEvent e) {
  if (e.getID() == WindowEvent.WINDOW_CLOSING) {
    cancel();
  }
  super.processWindowEvent(e);
}
```

saberf~1.jav



```
void cancel() {
   dispose();
}
public void actionPerformed(ActionEvent e) {
   if (e.getSource() == button1) {
      cancel();
   }
}
```







Saber to Bread Board Converter //Title: //Version: //Copyright: Copyright (c) 1998 James Geraci //Author: MIT LEES Lab //Company: //Description:Saber to Bread Board Converter package Thesis; import java.awt.*; import java.awt.event.*; import borland.jbcl.control.*; import borland.jbcl.layout.*; import java.io.*; import java.util.*; import java.text.*; public class SaberFrame extends DecoratedFrame { //Construct the frame BorderLayout borderLayout1 = new BorderLayout(); XYLayout xYLayout2 = new XYLayout(); BevelPanel bevelPanel1 = new BevelPanel(); MenuBar menuBar1 = new MenuBar(); Menu menuFile = new Menu(); MenuItem menuFileExit = new MenuItem(); Menu menuHelp = new Menu(); MenuItem menuHelpAbout = new MenuItem(); ButtonBar toolBar = new ButtonBar(); StatusBar statusBar = new StatusBar(); TextField textField1 = new TextField(); Button button1 = new Button(); TextField textField2 = new TextField(); TextField textField3 = new TextField(); Label label1 = new Label(); Label label2 = new Label(); TextField textField4 = new TextField(); Label label3 = new Label(); TextField textField5 = new TextField(); TextField textField6 = new TextField(); TextField textField7 = new TextField(); TextField textField8 = new TextField(); TextField textField9 = new TextField(); Label label4 = new Label(); Label label5 = new Label(); TextField textField10 = new TextField(); Label label6 = new Label();



```
TextField textField11 = new TextField();
 Label label8 = new Label();
 Label label9 = new Label();
 public SaberFrame() {
   try {
     jbInit();
   }
   catch (Exception e) {
     e.printStackTrace();
   }
 }
//Component initialization
 private void jbInit() throws Exception {
    this.setLayout(borderLayout1);
    this.setSize(new Dimension(466, 358));
    this.setTitle("Saber to BreadBoard Converter Program");
   menuFile.setLabel("File");
   menuFileExit.setLabel("Exit");
   menuFileExit.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e) {
        fileExit actionPerformed(e);
     }
   });
    menuHelp.setLabel("Help");
    menuHelpAbout.setLabel("About");
    menuHelpAbout.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e) {
        helpAbout_actionPerformed(e);
      }
    });
    toolBar.setButtonType(ButtonBar.IMAGE_ONLY);
    toolBar.setLabels(new String[] {"File", "Close", "Help"});
    textField1.setText("ece15_city.dat");
    button1.setLabel("GenerateFile");
    textField2.setText("0.594");
    textField3.setText("4.0");
    label1.setText("Tire Diameter :");
    label2.setText("Differential Gear Ratio :");
    textField4.setText("3.0");
    label3.setText("Engine-Alternator Gear Ratio :");
    textField5.setText("winter_worst_ece15.scs");
    textField6.setText("textField6");
    textField7.setText("BreadBoardCANLoads.txt");
```

99/06/12 20:08:06

textField8.setText("40"); textField9.setText("14"); label4.setText("Higher Voltage Bus Voltage :"); label5.setText("Lower Voltage Bus Voltage :"); textField10.setText("1800"); label6.setText("Programmable Load Wattage :"); textField11.setText("BBInputFile.txt"); label8.setText("e :"); label9.setText("Output FileName :"); button1.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(ActionEvent e) { button1_actionPerformed(e); } }); toolBar.setImageBase("image"); toolBar.setImageNames(new String[] {"openFile.gif", "closeFile.gif", "help.gif"}); bevelPanel1.setLayout(xYLayout2); menuFile.add(menuFileExit); menuHelp.add(menuHelpAbout); menuBar1.add(menuFile); menuBar1.add(menuHelp); this.setMenuBar(menuBar1); this.add(toolBar, BorderLayout.NORTH); this.add(statusBar, BorderLayout.SOUTH); this.add(bevelPanel1, BorderLayout.WEST); bevelPanel1.add(textField1, new XYConstraints(7, 20, 201, -1)); bevelPanel1.add(button1, new XYConstraints(349, 12, 99, 35)); bevelPanel1.add(textField2, new XYConstraints(388, 52, 60, 21)); bevelPanel1.add(textField3, new XYConstraints(388, 79, 60, 21)); bevelPanel1.add(label1, new XYConstraints(292, 52, -1, -1)); bevelPanel1.add(label2, new XYConstraints(249, 79, -1, -1)); bevelPanel1.add(textField4, new XYConstraints(388, 104, 60, 21)); bevelPanel1.add(label3, new XYConstraints(211, 102, -1, -1)); bevelPanel1.add(textField5, new XYConstraints(7, 52, 201, -1)); bevelPanel1.add(textField6, new XYConstraints(1, 244, 172, 34)); bevelPanel1.add(textField7, new XYConstraints(7, 79, 201, -1)); bevelPanel1.add(textField8, new XYConstraints(388, 129, 60, 21)); bevelPanel1.add(textField9, new XYConstraints(388, 157, 60, 21)); bevelPanel1.add(label4, new XYConstraints(220, 129, -1, -1)); bevelPanel1.add(label5, new XYConstraints(224, 156, -1, -1)); bevelPanel1.add(textField10, new XYConstraints(388, 185, 60, 21)); bevelPanel1.add(label6, new XYConstraints(208, 183, 155, 25)); bevelPanel1.add(textField11, new XYConstraints(7, 200, 161, -1)); bevelPanel1.add(label8, new XYConstraints(364, 183, 23, -1)); bevelPanel1.add(label9, new XYConstraints(7, 176, -1, -1));





```
}
//File | Exit action performed
 public void fileExit_actionPerformed(ActionEvent e) {
    System.exit(0);
  }
//Help | About action performed
  public void helpAbout_actionPerformed(ActionEvent e) {
    SaberFrame_AboutBox dlg = new SaberFrame_AboutBox(this);
    Dimension dlgSize = dlg.getPreferredSize();
    Dimension frmSize = getSize();
    Point loc = getLocation();
   dlg.setLocation((frmSize.width - dlgSize.width) / 2 + loc.x, (frmSize.height - dlgSize.height) / 2 + loc.y);
    dlg.setModal(true);
    dlg.show();
  }
void button1 actionPerformed(ActionEvent e)
  {
    textField6.setText("I have Started");
    Vector AlternatorInfoVector = new Vector(50000, 500);
    CANEventObject CANCollectorObject = new CANEventObject();
    /* Generate the Alternator RPM by reading in a *.dat file and then putting all of
    the data into a vecotr that contains AlternatorRPMObjects
      */
try{
    BufferedReader filein = new BufferedReader(new FileReader(textField1.getText()), 2000000);
    String s;
    while((s = filein.readLine()) != null)
    {
      s.trim();
      AlternatorInfoVector.addElement(new AlternatorRPMObject(textField2, textField3, textField4, s));
    }
     AlternatorInfoVector.trimToSize();
     filein.close();
  }
  catch(IOException ex)
  {
    System.exit(1);
  }
 /* Now Read in the SCS File and put the data into a Vector which contains
 objects of type CANEventObject*/
```



System.out.println("You are Starting the CANEventObjectFileHandler"); CANCollectorObject.CANEventObjectFileHandler(textField5.getText(), 0); System.out.println("You have completed the CANEventObjectFileHandler"); System.out.println("You are now starting the ReadinBreadBoardCANLoads"); /* The following function reads in a list of all known BreadBoardCANLoads*/ CANCollectorObject.ReadinBreadBoardCANLoads(textField7.getText()); /* The following function checks to see if the loads used in the Saber Simulation are Available on the CAN bus */ System.out.println("You are now starting the ConfirmBreadBoardCompatability"); CANCollectorObject.ConfirmBreadBoardCompatability(); /* The following function generates the serial messages which are to be used to activate the events on the CAN BUS it also puts them togther with their appropriate Alternator RPM Object*/ System.out.println("You are now starting the CreateCANMessages"); CANCollectorObject.CreateCANMessages(); /* The following function removes all the the Turn off Commands at t=0 */ System.out.println("You are now removing the excess turn off commands"); CANCollectorObject.removeTOoffMessages(); /* The following function generates the appropriate ElectroMechanical Valve Power Demand For a given Alternator Speed*/ System.out.println("You are now starting the GenerateEMValvePowerDemand"); CANCollectorObject.GenerateEMValvePowerDemand(textField8, textField9, textField10, AlternatorInfoVector); 11 Generate the Output File System.out.println("Now Writing the output file"); // int RunCounter = 0: // double NumberOfRunsDouble = new Double(textField13.getText()).doubleValue(); // int NumberOfRuns = (int) NumberOfRunsDouble;

try{

PrintWriter out = new PrintWriter(new FileOutputStream((textField11.getText()).trim()));

```
// while(RunCounter < NumberOfRuns)
// {
    int startupbuffer = 0;
    while(startupbuffer < 60)
        {
            out.println("//");
            startupbuffer++;
        }
</pre>
```



saberf~2.jav



```
int x = 0;
   //int CANEventListSize = CANCollectorObject.returnCANEventListSize();
   int z = AlternatorInfoVector.size() - 1;
   int peter = 0;
   double previousProgrammableLoad = 0;
   int previousRPM = -1;
   int fourthpoint = 1;
   int testRPM = 0;
   while (x < (z - 1))
    {
       System.out.println(x);
   11
    out.print("!" + x);
   // out.print(x);
    if(previousRPM != ((int) Integer.parseInt(((AlternatorRPMObject) (AlternatorInfoVector.elementAt(x))).getAlternatorShaft
Speed())))
        {
               testRPM = ((int) Integer.parseInt(((AlternatorRPMObject) (AlternatorInfoVector.elementAt(x))).getAlternatorSha
        11
ftSpeed()));
              testRPM = testRPM + 1;
        11
        // out.print("\t" + "?" + previousRPM);
        // System.out.println(previousRPM + " " + ((int) Integer.parseInt(((AlternatorRPMObject) (AlternatorInfoVector.eleme
ntAt(x))).getAlternatorShaftSpeed())));
         out.print("\t" + "?" + ((AlternatorRPMObject) (AlternatorInfoVector.elementAt(x))).getAlternatorShaftSpeed());
               previousRPM = ((int) Integer.parseInt(((AlternatorRPMObject) (AlternatorInfoVector.elementAt(x))).getAlternato
rShaftSpeed()));
     }
     //out.print("\t");
     peter = 0;
   while(peter < (CANCollectorObject.returnCANEventListSize()))</pre>
      {
        if((CANCollectorObject.returnCANEventTime(peter)) == x)
          {
           out.print("\t" + "#" + CANCollectorObject.returnCANString(peter));
          // CANCollectorObject.removeCANString(peter);
          }
          peter++;
      }
       if(previousProgrammableLoad != Double.valueOf(CANCollectorObject.returnProgrammableLoad(x)).doubleValue())
          {
            out.print("\t" + "^42+");
            out.print(CANCollectorObject.returnProgrammableLoad(x));
          }
```



}

11 }

}

{

}



```
previousProgrammableLoad = Double.valueOf(CANCollectorObject.returnProgrammableLoad(x)).doubleValue();
   if(fourthpoint == 1)
    {
    // These are the data collection CAN Calls.
    }
    else if(fourthpoint ==2)
    {
    out.print("\t" + "#A0030000BA0000000000000000BD0A");
    }
   /* out.print("\t" + "#A00300007000000000000000000A0A");
    out.print("\t" + "#A00300008000000000000000000B0A");
     }*/
   else if(fourthpoint == 3)
    {
     out.print("\t" + "#A0030000F0000000000000000120A");
    else if(fourthpoint ==4)
    {
    }
    else if(fourthpoint ==5)
    {
    fourthpoint = 0;
    }
    // out.print("\t" + "?" + ((AlternatorRPMObject) (AlternatorInfoVector.elementAt(x))).getAlternatorShaftSpeed());
    out.println("\t" + "//");
    fourthpoint++;
   x++;
// RunCounter++;
   out.close();
catch(IOException ex)
 System.exit(1);
 textField6.setText("I'm Done");
```



} } System.out.println("I'm Done");



sdr_locks 0001 sdr_driver 2001 sdr_turn 8001 sdr_brakes C001 sdr_abs_tc E002 sdr_defog 0016 sdr_heater 8003 sdr_rear_seat_htrs 0019 sdr_emissions 0004 sdr_windshield 4004 sdr_seat_htrs 0003

breadb~1.txt



- [1] Richard A. Perez, *The Complete Battery Book*, p. 134, Tab Books Inc, Blue Ridge Summit, PA 17214, first edition, 1985c1985, ISBN 0-83-6-0757-9.
- [2] E.E. Morton Arendt, Storage Batteries Theory, Manufacture, Care, and Application, p. 22, D. Van Nostrand Company, Inc., Eight Warren Street, New York, 1928c1928.
- [3] ACDelco 1999 Batteries Catalog 7A-100.
- [4] John Kassakian, "Automotive electrical systems circa 2005," *IEEE Spectrum*, 1997, http://auto.mit.edu/Consortia.nsf/ArticleViews.
- [5] Wolfhard Lawrenz, CAN System Engineering: From Theory to Practical Applications, Springer Verlag, 1997c1997, ISBN 0387949399.
- [6] Siemens AG, C167 Derivatives, 16-Bit CMOS Single-Chip Microcontroller, 2.0 edition, Apr.-May 1996, Section 8.
- [7] Siemens AG, C167 Derivatives, 16-Bit CMOS Single-Chip Microcontroller, 2.0 edition, Apr.-May 1996, Section 16.
- [8] Irene Kuo, "A methodology for sizing components in a dual-voltage automotive electrical system," Tech. Rep., Massachusetts Institute of Technology, 1999.
- [9] James K. Roberge, Operational Amplifiers, Theory and Practice, p. 458, John Wiley & Sons, Inc., New York, 1975c1975, ISBN 0-471-72585-4.