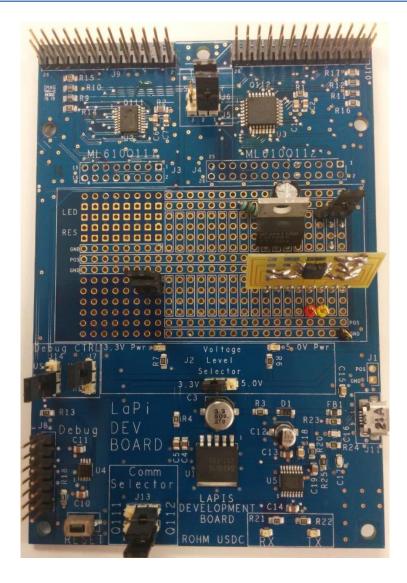


BD99935 I2C Emulator Board Manual



Above: Lapis Development Kit with added EEPROM and LDO for supporting BD99935 pre-testing

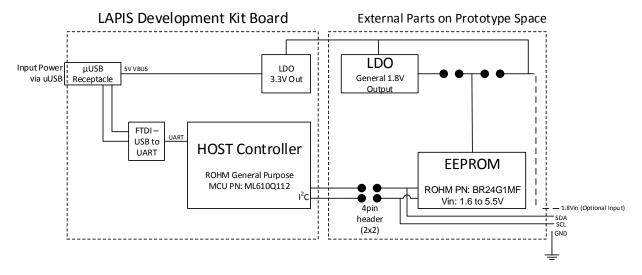


Table of Contents

evel Block Diagram	3
Start Guide	3
DM Register Contents	4
vare Board Explanation	5
nper Explanations	5
mary Hardware Points	6
vare Flow Chart	7
h Level Flowchart of Operation	7
ROM Register Content Confirmation	8
Nethod 1: Checking the EEPROM using the on-board I2C Reads	8
Nethod 2: Checking the FEPROM using an Aardvark	10



High Level Block Diagram



• The High level block diagram for this application board can be seen above. The left side of this picture shows the blocks used on the existing Lapis Development Kit. The right side of this picture shows the block used on the prototype space of the Lapis Development Kit.

Quick Start Guide

- 1. The jumpers should be set when we send the board, but jumpers should match the board as per the definitions in the "Hardware Board Explanations" section
- 2. Power the board using the a uUSB cable to Connector J11
- 3. Upon power-on the EEPROM will be initialized with the appropriate register map
 - o Note: the RESET button can be used to re-initialize the EEPROM register map
- 4. When ready to connect to the different host, please disconnect the I2C pin jumpers to the Host MCU to ensure the I2C data bus is free.
- 5. Also, depending on the voltage level, adjust the jumper above the LDO
 - LEFT most Pins = 1.8V
 - o RIGHT most Pins = 3.3V
- 6. Then connect the pins to the board
 - Orange = SDA
 - Yellow = SCL
 - o Black = GND

EEPROM Device Address (7bit) = 0x50

IMPORTANT NOTE: This EEPROM requires a TWO BYTE register addressing scheme.



EEPROM Register Contents

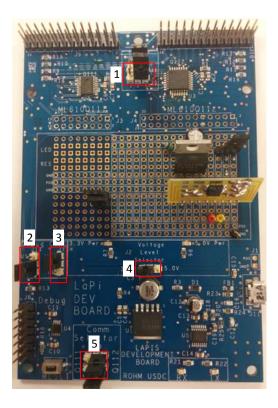
Addres		R/									Initial	
	Reg Name	W	D7	D6	D5	D4	D3	D2	D1	D0	Value	
	VENDORID	R				VEND	ORID[7:0]					Defined
0x01	REVID	R		MAJRE	EV[3:0]			MINRE	EV[3:0]		0xA0	Defined
		R/										
0x02	IRQLVL1	W	-	-	-	-	-	ADC	PWRSRC	TEMP	0x00	Level1 interrupt register
		R/										
0x03	IRQLVL1M	W	-	-	-	-	-	ADCM	PWRSRCM	TEMPM		Level1 interrupt mask register
		R/					CRITTEMPO	WARNTEMP	CRITTEMPEX	WARNTEMPE		
0x04	IRQTEMP	W	-	-	-	-	D	OD	T	XT	0x00	Temperature Second Level interrupt register
		R/					OVPWARND					
0x05	IRQPWRSRC	W	-	IRECTOCP	WC READY	LMDONE	ET	OCVRECT	UVVRECT	OVVRECT	0x00	Power Source Second Level interrupt register
		R/							ADCVRECTIO			
0x06	IRQADC	W	-	-	-	-	-	ADCMBICAL	AL	RND		ADC Second Level interrupt register
		R/					CRITTEMPO	WARNTEMP		WARNTEMPE		
0x07	IRQTEMPM	W	-	-	-	-	DM	ODM	TM	XTM	0x0F	Temperature Second Level interrupt mask register
		R/			WC READY		OVPWARND					
0x08	IRQPWRSRCM	W	-	IRECTOCPM	_M	LMDONEM	ETM	OCVRECTM	UVVRECTM	OVVRECTM	0x4E	Power Source Second Level interrupt mask register
		R/						ADCMBICAL	ADCVRECTIC			
0x09	IRQADCM	W	-	-	-	-	-	M	ALM	RNDM	0x07	ADC Second Level interrupt mask register
		R/										
0x0A	VRCNT1	W	LDOEN	LDOPGEN	_	_	_	_	_	_	0xC0	Voltage Regulator Control Register1
		R/										
0x0B	VRCNT2	W	SBEN	SBPGEN	_	_	_	_	WCR EN	_	0xC2	Voltage Regulator Control Register2
		R/										
0x0C	OVPWARN	W		OVP S	ET[3:0]		_	(OVPWARN[2:0)]	0x00	OVP Warning Register
		R/										
0x0F	LDMOD	W	-	_	_	_	_	_	WCOUTEN	LMEN	0x01	Load Modulation / Simple Buck Driver control register
		R/		MBICALOFFS								
0x30	ADCCNT1	w	_	ETEN	MBICALEN	ADEN	ADSTRT		ADSLP[2:0]		0x18	ADC Control register1
		R/		VRECTBITSE								
0x31	ADCCNT2	w	_	1	N	ETEN	N	ADCTHERM		RRDATARD	0x00	ADC Control register2
0,101	/ LD G G I I I I	R/						7.00111.011		THE CONTINUES	ONOO	ab o donaro regionare
0x33	ADCADDR0	w	_			AVRGCH0	STOPCH0		ADSEL0[2:0]		0x00	ADC Address Register 0
OXOG	ADOADDIKO	R/				AVICOGIIO	01010110		ADOLLO[2.0]		OXOO	ADO Address register o
0v34	ADCADDR1	l w	_	_		AVRGCH1	STOPCH1		ADSEL1[2:0]		0200	ADC Address Register 1
0.0.0-1	ADOADDIKI	R/				AVICOUIT	01010111		ADOLL 112.01		OXOO	ADO Address register i
0v35	ADCADDR2	w	_	_		AVRGCH2	STOPCH2		ADSEL2[2:0]		0v00	ADC Address Register 2
OXOO	POUNDOILE	R/				AVIOUIL	01010112		ADOLLZ[Z.0]		0,000	nDO Mudicas Register 2
0×36	ADCADDR3	w				AVRGCH3	STOPCH3		ADSEL3[2:0]		0200	ADC Address Register 3
0,000	MDOMDDIAG	R/	-	-		AVINOUNS	3101013		ADSELS[Z.0]		0000	ADO Address Negister 5
0.27	ADCADDR4	W				AVRGCH4	STOPCH4		ADSEL4[2:0]		٥٠٠٥	ADC Address Register 4
UX3/	ADCADUK4	R/	-	-	-	AVRUUH4	310PCH4		MUSEL4[2:0]		UXUU	MDO Address register 4
0.20	ADCADDR5	W I				AVRGCH5	STOPCH5		ADSEL5[2:0]		٥٠٠٥	ADC Address Register 5
UXOG	MDOMDDUS	VV	-	-	-	AVROUND	310F0H5		AUSELU[Z.U]		UXU8	NDO Address Negister 0

		-										· · · · · · · · · · · · · · · · · · ·
Addres		R/		l			l I				Initial	
S	Reg Name	W	D7	D6	D5	D4	D3	D2	D1	D0	Value	
	ADCVRECTOFFSE											
0x40	TL	R				ADCVREC	TOFFSETL[7:0]				0x00	ADC Offset Data Register for VRECT Current Calibration(Low)
	ADCVRECTOFFSE											
0x41	TH	R	-	-	-	-	-	-	ADCVRECTO	DFFSETH[1:0]	0x00	ADC Offset Data Register for VRECT Current Calibration(High)
0x42	ADCMBIOFFSETL	R				ADCMBIO	OFFSETL[7:0]				0x00	ADC Offset Data Register for Main Buck Current Calibration(Low)
0x43	ADCMBIOFFSETH	R	-	-	-	-	-	-	ADCMBIOF	FSETH[1:0]	0x00	ADC Offset Data Register for Main Buck Current Calibration(High)
		R/										
0x44	ADCCRTTMPEXTL	w				ADCCRIT	TMPEXTL[7:0]				0x26	ADC crirical external temp-sensor threshold level register(Low)
	100011111111111111111111111111111111111	R/				7.0001111					OTILLO	as a string a strong series an assistant of register (2017)
0v45	ADCCRTTMPEXTH	W	_	l .				_	ADCCRITTA	MPEXTH[1:0]	0200	ADC crirical external temp-sensor threshold level register(High)
OX 10	ADOORT TIME EXTITE	R/							7.000111111	iii EXTITITI.O	OXOO	ADC warning level external temp-sensor threshold level
0v46	ADCWRNTMPEXTL	W				ADCW/ARI	NTMPEXTLI7:01				0v51	register(Low)
0.40	ADCWRNTMPEXT	R/				ADOWAIN	TIWI EXTER				0.001	ADC warning level external temp-sensor threshold level
0x47	LI LI	W							ADCIMADNIT	MPEXTH[1:0]	0400	register(High)
UX47	П	R/		-	_	_		_	ADCWARNI	MPEXITION		ADC VRECT output load current threshold level register for MSB
	ADDOOUSEDT!	W					VDE07/7.01				0x28	ADC VRECT output load current infeshold level register for WSB
0x48	ADCOCVRECTL					ADCOC	VRECT[7:0]				0x28	DDIT
1	l		IRECTOCP									
0x49	IRECTOCP	W	DIS	OCVRECTOR	-	-	-	-	I IRECTO	DCP[1:0]		VRECT input current over current protection level setting
1												ADC VRECT dynamic range setting Read Only. Read data
0x4A	VRANGESET1	R	-	-	-	-	-	VI	RANGESET1[2	:0]	0x	depend on WC READY
1		R/										
0x4B	VRANGESET2	W	WRITEEN	-	-	-	-	-	VRANGE:	SET2[1:0]	0x00	ADC VRECT dynamic range setting
		R/										
	LVLDETIREF	W	-	-	-	-		-	LVLDETI	REF[1:0]		LVLDET block iref setting
0x50	ADCSNS0L	R				ADCS	SNS0L[7:0]				0x00	ADC Round Robin Data0(Low)
0x51	ADCSNS0H	R				ADCS	SNS0H[7:0]				0x00	ADC Round Robin Data0(High)
0x52	ADCSNS1L	R				ADCS	SNS1L[7:0]				0x00	ADC Round Robin Data1(Low)
	ADCSNS1H	R					SNS1H[7:0]					ADC Round Robin Data1(High)
	ADCSNS2L	R					SNS2L[7:0]					ADC Round Robin Data2(Low)
	ADCSNS2H	R					NS2H[7:0]					ADC Round Robin Data2(High)
	ADCSNS2H ADCSNS3L	R					SNS3L[7:0]					ADC Round Robin Data3(Low)
	ADCSNS3H	R					SNS3H[7:0]					ADC Round Robin Data3(Low)
	ADCSNS3H ADCSNS4L	R					SNS3H[7:0] SNS4L[7:0]					ADC Round Robin Data3(High) ADC Round Robin Data4(Low)
	ADCSNS4H	R					SNS4H[7:0]					ADC Round Robin Data4(High)
	ADCSNS5L	R					SNS5L[7:0]					ADC Round Robin Data5(Low)
0x5B	ADCSNS5H	R				ADCS	NS5H[7:0]				0x00	ADC Round Robin Data5(High)
		R/		l			I I					
0x60	UVLOVINDB	W	-	-	-	-	-	l	JVLOVINDB[2:0	0]	0x00	UVLO Debounce Time Select
		R/										
0x61	SMBDBG	W	-		-			-	SMBCPCL	KSEL[1:0]	0x00	Simple Buck Control Register
		R/										
0x62	WCWAIT	W	_	-	_	-	.	-	WCWAIT	TSEL[1:0]	0x00	After WC READY Wait Time Control Register
		R/	SOFTRESE									
0x70	SOFTRESET	W	T	1							0x00	Soft Reset Control Register
OATO	O O . IIKEOLI										0000	post record outside integrated



Hardware Board Explanation

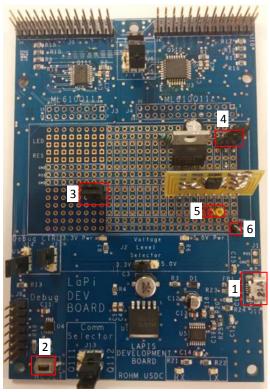
Jumper Explanations



- 1. This is a selector Jumper for choosing which MCU to be programmed. For this application this is N/A since the user will not be re-programming the MCU.
 - a. J5 = Jump Pins 1 and 2
 - b. J6 = Jump Pins 1 and 2
- 2. This is a selector jumper to choosing how to connect the RESET pin. For this application, we want to use the RESET push button to reset the application if an EEPROM re-write is required.
 - a. J14 = Jump USR to RST
- 3. This is a selector jumper for choosing how to source power to this board. For this application we want to source 5V power from the uUSB connector.
 - a. J7 = Jump USR to PWR
- 4. This is a selector jumper for choosing the voltage level on the "POS" lines of the prototyping space and MCU source power. For this application can use either 3.3V or 5V (but tested under 3.3V condition)
 - a. J2 = Jump 3.3V to middle pin
- 5. This is a selector jumper for choosing the output MCU for the UART lines for either the Q111 or Q112 MCU. This function is not used in this application
 - a. Default: Connect Q112 side to middle pin for both jumpers



Primary Hardware Points

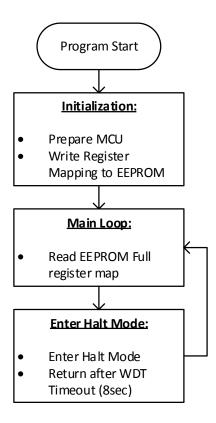


- 1. uUSB Receptacle
 - a. This serves two functions
 - i. Power the board using the USB 5V VBUS line
 - ii. Sets up a UART debug COM port on the host PC
 - b. For this application, only 5V Vbus is used
- 2. Reset Button for the LAPIS Q112 MCU
 - a. For this application, this will serve to re-write the EEPROM to the original settings
- 3. I2C Connection Jumpers
 - a. This connects the I2C bus lines to the onboard LAPIS Q112 MCU.
 - b. When connecting to an external host, these should not be jumped
 - c. When resetting the EEPROM memory map, this should be jumped
- 4. On-Board Power Selector
 - a. This connector allows the user to select the voltage rail of the EEPROM.
 - i. Jump Left = 1.8V
 - ii. Jump Right = 3.3V
- 5. External Jumpers for I2C Connection
 - a. Orange = SDA
 - b. Yellow = SCL
- 6. External Jumper for GND



Firmware Flow Chart

High Level Flowchart of Operation



- The Purpose of the MCU on this board is to complete only 1 task: to write the default register mapping to the EEPROM device.
- In the main loop, the EEPROM is read to check the register contents, but ultimately is not helpful to the end user.



EEPROM Register Content Confirmation

In order to ensure the EEPROM is working as specified, we confirmed operation of this device using two methods.

Method 1: Checking the EEPROM using the on-board I2C Reads

```
👺 main.c
                                                                                                  Line BP Address
                 #define ESC_NEWLINE
                #define ESC_PREVLINE "\033[F"
 00211
                #define ESC_ERASE2END "\033[J"
 00212
 00213
00214
               // Start of MAIN FUNCTION
                //-----
 00215
         int main(void)
 00216
 00217
 00218 🗆 0:0AC6H
                        Initialization(); //Ports, UART, Timers, Oscillator, Comparators, etc.
 00219
                      #ifdef DebugOn
 00220 🔲 0:0ACAH
                       PRINTF("Start Program");
 00221
 00222 🛘 0:0ADAH
                      I2C Write(BR24 I2C ADDR, &BR24 REG00, 2, &BR24 REG00 Contents, 255);
 00223
 00224
                MainLoop:
                       main_clrWDT();
 00225 U 0:0AF4H
 00226
 00227 U 0:0AF8H
                      I2C_Read(BR24_I2C_ADDR, &BR24_REG00, 2, &Test00_Return, 255);
 00228
00229 U 0:0B12H
                      HLT = 1;
                                      //Wait time here depends on the WDT timing
                       __asm("nop\n");
00230  0:0B16H
00231  0:0B18H
                        __asm("nop\n");
 00232
                       goto MainLoop;
00234
 00235
                    End of MAIN FUNCTION
 00236
 00237
 00238
 00239
                //-----
 00240
 00241
                 // Start of Other Functions...
 00242
 00243
 00244
                // Initialize Micro to Desired State...
 00245
                 //====
 00246 U 0:0CE2H static void Initialization(void) {
 00247
 00248
                       //Initialize Peripherals
 00249
                       //BLKCON2 Control Bits...Manually Set 4/12/2013
 00250 U 0:0CE4H
                       DSIO0 = 1; // 0=> Enables Synchronous Serial Port 0 (initial value).
 00251
                       #ifdef DebugOn
 00252 U 0:0CE8H
                       DUA0 = 0; // 0=> Enables the operation of UARTO (initial value).
 00253
                        #endif
00254
                        #ifndef DebugOn
00255
                       DUA0 = 1; // 0=> Enables the operation of UARTO (initial value).
 00256
                        #endif
                        DUA1 = 1: // 0=> Enables Hart1 (initial value)
```

In the above picture, we have setup a breakpoint after the MCU goes into Halt Mode. Thus, at this point, the MCU has written and re-read the register contents of the EEPROM. The Contents are then placed in the character array "Test00_Return". Upon checking this variable in the debugger, we can see that the register values were taken successfully



Expression	Value	Type Memory	y Address
Test00		unsig	
[0]	0x1F	unsig RAM	00:EEDCH
··· [1]	0xA0 ' '	unsig RAM	00:EEDDH
··· [2]	0x0	unsig RAM	00:EEDEH
··· [3]	0x7	unsig RAM	00:EEDFH
··· [4]	0x0	unsig RAM	00:EEEOH
··· [5]	0x0	unsig RAM	00:EEE1H
··· [6]	0x0	unsig RAM	00:EEE2H
··· [7]	0xF	unsig RAM	00:EEE3H
[8]	0x4E 'N'	unsig RAM	00:EEE4H
[9]	0x7	unsig RAM	00:EEE5H
··· [10]	0xC0 'À'	unsig RAM	00:EEE6H
··· [11]	0xC2 'Â'	unsig RAM	00:EEE7H
[12]	0x0	unsig RAM	OO:EEE8H
[13]	0x0	unsig RAM	00:EEE9H
[14]	0x0	unsig RAM	00:EEEAH
[15]	0x1	unsigRAM	00:EEEBH
[16]	0x0	unsig RAM	00:EEECH
[17]	0x0	unsig RAM	00:EEEDH
[18]	0x0	unsig RAM	OO:EEEEH
[19]	0x0	unsigRAM	OO:EEEFH
	0x0	-	OO:EEFOH
··· [20]		unsig RAM	00:EEF1H
[21]	0x0	unsig RAM	
[22]	0x0	unsig RAM	00:EEF2H
[23]	0x0	unsigRAM	00:EEF3H
[24]	0x0	unsigRAM	00:EEF4H
[25]	0x0	unsigRAM	00:EEF5H
[26]	0x0	unsig RAM	00:EEF6H
[27]	0x0	unsig RAM	00:EEF7H
[28]	0x0	unsig RAM	00:EEF8H
··· [29]	0x0	unsig RAM	00:EEF9H
··· [30]	0x0	unsig RAM	00:EEFAH
··· [31]	0x0	unsig RAM	00:EEFBH
··· [32]	0x0	unsig RAM	00:EEFCH
··· [33]	0x0	unsig RAM	00:EEFDH
··· [34]	0x0	unsig RAM	00:EEFEH
··· [35]	0x0	unsig RAM	00:EEFFH
··· [36]	0x0	unsig RAM	00:EF00H
··· [37]	0x0	unsig RAM	00:EF01H
··· [38]	0x0	unsig RAM	00:EF02H
[39]	0x0	unsig RAM	00:EF03H
··· [40]	0x0	unsig RAM	00:EF04H
[41]	0x0	unsig RAM	00:EF05H
[42]	0x0	unsig RAM	00:EF06H
[43]	0x0	unsig RAM	00:EF07H
··· [44]	0x0	unsig RAM	00:EF08H
[45]	0x0	unsig RAM	
atch 1 Watch		unsig KAN	00.EF03R



Method 2: Checking the EEPROM using an Aardvark

In the below picture, the Total Phase Aardvark I2C tool was used to confirm the operation of this device



After Performing the full register map read, we can see that the contents are loaded successfully.

