

INTRODUCTION TO PROGRAMMING PIC32

January 12, 2010

Andy Long

Special Function Registers

- ▣ Ex. TRISX = 0x0000
- ▣ More understanding of underlying code
- ▣ More difficult to program
- ▣ Microchip Reference Manual
- ▣ Jasio book – Programming 32-bit Microcontrollers in C

Macros / Functions

- ▣ Ex. ReadADC10(bufIndex)
- ▣ SFR $\rightarrow$ $(*(\&\text{ADC1BUF0} + ((\text{bufIndex}) * 4)))$
- ▣ Special Function Registers are used underneath the function (can be seen in the header and source files)
- ▣ PIC32 Peripheral Libraries for MPLAB C32 Compiler

Outline

- ▣ Digital Inputs / Outputs – SFR
- ▣ Analog Inputs – Macros / Functions

SFR – I/O Ports

- ▣ TRISx – initializes pins as input or output
- ▣ PORTx – digital input
- ▣ LATx – digital output
- ▣ ODCx – open drain (similar to transistor)
- ▣ CNCON, CNEN, CNPUE – change notification

Available Pins

Port	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
A									X	X		X			X	
B																
C	X			X												
D			X	X					X	X	X					
E									X	X	Out	Out	Input	Input	Input	Input
F													X		X	X
G	X	X	X	X									X	X	X	X

PIC32

PIC32 but not available
x for programming

Not on PIC32

- TRIS_x, PORT_x, and LAT_x operate on a 16 bit number for port x
- Each bit corresponds to a given pin
- Note: not always 16 pins available for each port
- Note: port B defaults to analog input

TRISx

- ▣ TRISx bit = 0 → sets up digital output
- ▣ TRISx bit = 1 → sets up digital input
- ▣ TRISx &= 0xFFFF0
 - TRISx &= 1111 1111 1111 0000
 - Makes bits 0, 1, 2, 3 digital outputs
- ▣ TRISx |= 0x00F0
 - TRISx |= 0000 0000 1111 0000
 - Makes bits 4, 5, 6, 7 digital inputs
- ▣ TRISEbits.TRISE0 = 0
 - Makes E0 digital output
- ▣ Functions
 - PORTSetPinsDigitalOut(IOPORT_C, IOPORT_PIN_6 | IOPORT_PIN_7);

PORTx (Input)

- ▣ Read High or Low for pin / port (READ)
- ▣ #define PIN_A9 PORTAbits.RA9
 - value = PIN_A9
- ▣ Functions:
 - unsigned int mPORTxRead(void)
 - unsigned int mPORTxReadBit(unsigned int _bits)

LATx (Output)

- ▣ Sets pin / port as High or Low (Write)
- ▣ `#define PIN_D1 LATDbits.LATD1`
 - `PIN_D1 = 1; // Sets High`
 - `PIN_D1 = 0; // Sets Low`
- ▣ Functions:
 - `void mPORTxWrite(unsigned int _value)`

Analog Input

- ▣ 10 Bit Analog to Digital Conversion (ADC)
- ▣ 0 – 1023 for (0 to 3.3 V)

- ▣ Examples
 - Potentiometer
 - Accelerometer

17.2 CONTROL REGISTERS

The ADC module includes the following Special Function Registers (SFRs):

The AD1CON1, AD1CON2 and AD1CON3 registers control the operation of the ADC module.

- AD1CON1: ADC Control Register 1
AD1CON1CLR, AD1CON1SET, AD1CON1INV: Atomic Bit Manipulation, Write-only Registers for AD1CON1.
- AD1CON2: ADC Control Register 2
AD1CON2CLR, AD1CON2SET, AD1CON2INV: Atomic Bit Manipulation, Write-only Registers for AD1CON2.
- AD1CON3: ADC Control Register 3
AD1CON3CLR, AD1CON3SET, AD1CON3INV: Atomic Bit Manipulation, Write-only Registers for AD1CON3.

The AD1CHS register selects the input pins to be connected to the SHA.

- AD1CHS: ADC Input Channel Select Register
AD1CHSCLR, AD1CHSSET, AD1CHSINV: Atomic Bit Manipulation, Write-only Registers for AD1CHS.

The AD1PCFG register configures the analog input pins as analog inputs or as digital I/O.

- AD1PCFG: ADC Port Configuration Register
AD1PCFGCLR, AD1PCFGSET, AD1PCFGINV: Atomic Bit Manipulation, Write-only Registers for AD1PCFG.

The AD1CSSL register selects inputs to be sequentially scanned.

- AD1CSSL: ADC Input Scan Selection Register
AD1CSSLCLR, AD1CSSLSET, AD1CSSLINV: Atomic Bit Manipulation, Write-only Registers for AD1CSSL.

The ADC module also has the following associated bits for interrupt control:

- Interrupt Request Flag Status bit (AD1IF) in IFS1: Interrupt Flag Status Register 1
- Interrupt Enable Control bit (AD1IE) in IEC1: Interrupt Enable Control Register 1
- Interrupt Priority Control bits (AD1IP<2:0>) and (AD1IS<1:0>) in IPC6: Interrupt Priority Control Register 6

Bit Registers

- ▣ AD1CON1 = 1010 0000 1110 0100

- ▣ OpenADC10(config1, config2, config3,
configport, configscan)
 (mPORTBSetPinsAnalogIn(configport),
 AD1CSSL = ~(configscan),
 AD1CON3 = (config3),
 AD1CON2 = (config2),
 AD1CON1 = (config1))

- ▣ Macros / Functions are a lot easier to read and understand

AD1CON1

Register 17-1: AD1CON1: ADC Control Register 1

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 31				bit 24			

r-x	r-x	r-x	r-x	r-x	r-x	r-x	r-x
—	—	—	—	—	—	—	—
bit 23				bit 16			

R/W-0	R/W-0	R/W-0	r-x	r-x	R/W-0	R/W-0	R/W-0
ON	FRZ	SIDL	—	—	FORM<2:0>		
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	r-x	R/W-0	R/W-0	R/C-0
SSRC<2:0>			CLRASAM	—	ASAM	SAMP	DONE
bit 7				bit 0			

PIC32MX460F512L.h

C:\Program Files \Microchip\MPLAB C32 \
pic32mx\include\proc\pic32mx460f512l.h

- ▣ _AD1CON1_ADON_POSITION = 15
- ▣ _AD1CON1_ADSIDL_POSITION = 13
- ▣ _AD1CON1_FORM_POSITION = 8
- ▣ _AD1CON1_SSRC_POSITION = 5
- ▣ _AD1CON1_ASAM_POSITION = 2

AD1CON1

1000 0000 0000 0000

0000 0000 0000 0000

0000 0000 1110 0000

0000 0000 0000 0100

1000 0000 1110 0100

Example Code

- ▣ C:\Program Files\Microchip\MPLAB
C32\examples\plib_examples\adc10\adc10_b
asic\source

Lab

- ▣ PCB
- ▣ Digital Input / Output
- ▣ Analog Input
- ▣ LCD Display
- ▣ Computer Communication – RS232