

Microcontroller Introduction

ECE110

Lecture 1

John A. Chandy

Adapted from Prof. Martin Fox's ECE266 Notes



University of
Connecticut



Agenda

- Course outline/ goals and approach
- Quick Overview of Computer Architecture
- Definition of Microcontrollers
- Components
- Microcontroller Market
- Key Suppliers



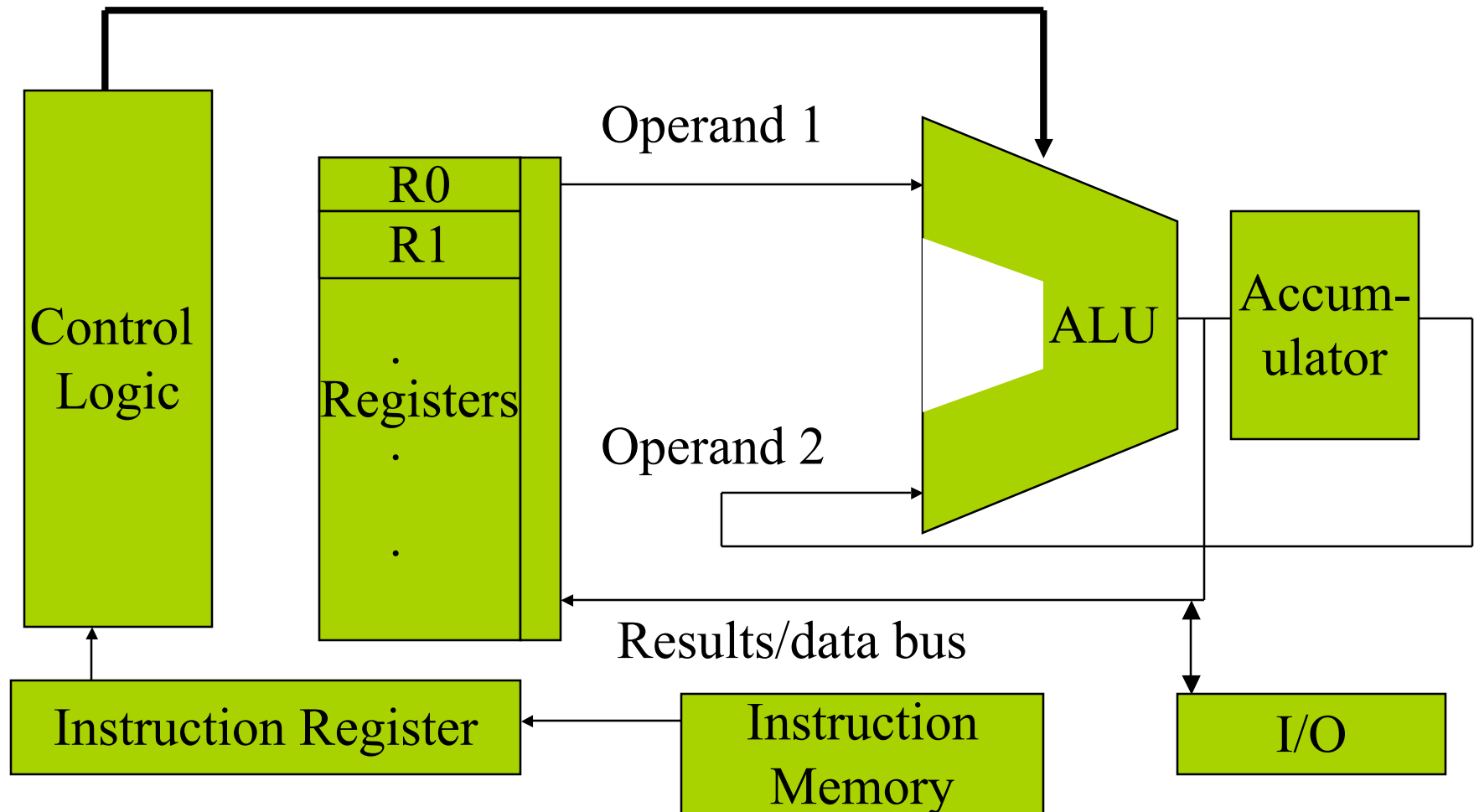
Introduction

- Basics of computer architecture
- How does a computer work?
- Microcontrollers, microprocessors and microcomputers
- Microcontrollers are general purpose digital logic replacements

Course goals

- Learn about microcontrollers
- Learn how to apply them in engineering applications
- Learn the basics of embedded systems
- Provide framework for future design lab courses

Basic Computer Architecture



RISC Vs. CISC

- RISC = Reduced Instruction Set Computer
- CISC = Complex Instruction Set Computer
- Microcontrollers = mostly RISC
- Microcomputer for PC = x86 is CISC
- Pipelining = easier in RISC
- PIC = 35 instructions



Examples of Instructions: 1

ADDLW	Add Literal and W
Syntax:	<code>[label] ADDLW k</code>
Operands:	$0 \leq k \leq 255$
Operation:	$(W) + k \rightarrow (W)$
Status Affected:	C, DC, Z
Description:	The contents of the W register are added to the eight bit literal 'k' and the result is placed in the W register.

ANDWF	AND W with f
Syntax:	<code>[label] ANDWF f,d</code>
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
Operation:	$(W) .AND. (f) \rightarrow (\text{destination})$
Status Affected:	Z
Description:	AND the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Immediate: Add literal and w Register: AND W with F

From: PIC data sheet

Assembly Language

- Human readable coding of machine language
- Assembler translates to binary
- Each line-> one instruction



Example of assembly Language program

```
p1874
: P1 PROGRAM :
: Toggle the top three LEDs every half second in sequence: bottom, middle, top.
: Use 4 MHz crystal for 1 microsecond internal clock period.
: Use Timer2 to obtain ten millisecond loop time.
: Toggle bit 5 of PORTA each time around the mainline loop (to test loop time).
: Echo RPG "encoder emulator" outputs to the bottom two LEDs in response to
: presses of the "INC" and "DEC" pushbuttons.
:
: Program hierarchy :
Mainline
: Initial
: Blink
:   BlinkTable
:   LoopTime
: IntService
:   RPG
:
:
: list P=PIC16F874, F=INHX8M, C=160, N=77, ST=OFF, MM=OFF, R=DEC, X=OFF
: #include P16F874.inc
: __config(_CP_OFF & _PWRTE_ON & _XT_OSC & _WDT_OFF & _BODEN_OFF)
:
: Equates :
Bank0RAM equ    H'20'           ;Start of Bank 0 RAM area
MaxCount equ    50             ;Number of loops in half a second
:
: Variables :
cblock Bank0RAM
W_TEMP          ;Temporary storage for w during interrupts
STATUS_TEMP     ;Temporary storage for STATUS during interrupts
BLNKCNT         ;Loop counter for blinking LEDs every half sec.
TEMP            ;Temporary variable for BlinkTable subroutine
endc
:
: Macro definitions :
MOVLW macro    literal,dest
movlw    literal
movwf   dest
endm

MOVFF macro    source,dest
movf    source,w
movwf   dest
endm
:
: Vectors :
org     H'000'           ;Reset vector
nop                    ;'no operation'; needed for in circuit debugger
goto   Mainline        ;Branch past tables
org     H'004'           ;Interrupt vector
goto   IntService      ;Branch to interrupt service routine
:
: BlinkTable subroutine :
: This subroutine reads PORTD and retains only the upper three LED bits. It
```

```

BlinkTable
MOVFF PORTD,TEMP ;Copy present state of LEDs to TEMP
swpff TEMP,F ;Move bits 7,6,5 to bits 3,2,1
rrf TEMP,W ; and on to bits 2,1,0 of W
andlw B'00001111' ;Keep only bits to be shifted
addwf PCL,F ;Change PC with PCLATH and offset in W
retlw B'00100000' ;(000 -> 001) reinitialize to bottom
retlw B'01100000' ; 001 -> 010 bottom to middle
retlw B'11000000' ; 010 -> 100 middle to top
retlw B'01000000' ;(011 -> 001) reinitialize to bottom
retlw B'10100000' ; 100 -> 001 top to bottom
retlw B'10000000' ;(101 -> 001) reinitialize to bottom
retlw B'11100000' ;(110 -> 001) reinitialize to bottom
retlw B'11000000' ;(111 -> 001) reinitialize to bottom

;:::::: End of Tables ;::::::::::::::::::::::::::::::::::::::::::::::::::
;:::::: Mainline program ;::::::::::::::::::::::::::::::::::::::::::::::::::

Mainline
call Initial ;Initialize everything
MainLoop
call Blink ;Blink upper three LEDs
call LoopTime ;Force loop time to be ten milliseconds
goto MainLoop

;:::::: Initial subroutine ;::::::::::::::::::::::::::::::::::::::::::::::::::
;
; This subroutine performs all initializations of variables and registers.
Initial
bsf STATUS,RP0 ;Set register access to bank 1
MOVLFB'00000100',ADCON1 ;select PORTA pins for ADC or digital I/O
MOVLFB'00001011',TRISA ;Set I/O for PORTA
MOVLFB'00001111',TRISB ;Set I/O for PORTB
MOVLFB'11010111',TRISC ;Set I/O for PORTC
clrf TRISD ;Set I/O for PORTD
MOVLFB'00000100',TRISE ;Set I/O for PORTE
MOVLFB'10,PR2 ;Set up Timer2 for a looptime of 10 ms

bcf STATUS,RP0 ;Set register access back to bank 0
MOVLFB'01001101',T2CON ;Finish set up of Timer2 (see page 62)
clrf PORTD ;Turn off LEDs
MOVLFB'11010000',INTCON ;Enable RB0/INT interrupts (see page 98)
return

;:::::: Blink subroutine ;::::::::::::::::::::::::::::::::::::::::::::::::::
;
; This subroutine blinks a new LED every 0.5 second.
Blink
decfsz BLNKCNT,F ;Decrement loop counter and return if not zero
goto BlinkEnd
movlw MaxCount ;Reinitialize BLNKCNT
movwf BLNKCNT
call BlinkTable ;Get bits to change into W
xorwf PORTD,F ; and toggle them into PORTD
BlinkEnd
return

;:::::: LoopTime subroutine ;::::::::::::::::::::::::::::::::::::::::::::::::::
;

```



Higher Level Languages

- Compiler translates from code to computer instruction set.
- Examples: Basic, Fortran, C, C++, Java
- Application packages: Mathematica, Mathcad, spreadsheets -> even higher level languages



Operating Systems

- Perform I/O I.e. printing, storage to disk
- Graphical user interface [GUI], I.e. 'Windows'
- Virtual memory
- Resource sharing [allows multiple programs to be running at the same time]
- Network interface
- Security



Embedded Systems

- Dedicated computers for 'SMART' applications
- Automotive: engine control, transmission control
- Microwaves, CDs, cellular telephones, remotes
- DSP: embedded controller optimized for math operations [multiplies, etc.] often used for image and sound processing



Summary

- We have given you a brief survey of computer and microprocessor essentials
- Computer engineers use microprocessors as part of larger devices, I.e. electronic and control systems [embedded computers]
- Such microprocessors can act as logic replacement devices
- Microprocessor-based systems are easier to document and upgrade because the design is in the [changeable] coding



Microcontrollers vs. Microprocessors

- What is the difference between a microprocessor and a microcontroller?
- A microcontroller usually incorporates other specialized components that are useful in embedded systems
 - Serial ports (RS-232, USB)
 - Networking (WiFi, Ethernet, ZigBee)
 - On-board memory (Flash, DRAM, SRAM)
 - Audio (MP3 encoding, signal processing)
 - Analog I/O (DAC, ADC)



Microcontrollers Overview

- Microcontrollers - a key impact technology for the 21st century
- Microcontrollers.com: “In the aggregate, PC microprocessors are responsible for less than 1% of all processors sold. Embedded processors outsell PC processors by more than 99%.”
- This course will provide enough information AND practical experience to get you started on the road to developing your own designs

Microcontrollers and Embedded Controllers

- Controls some process or aspect of the environment: [Microcontrollers Vs. DSPs](#)
- DSPs optimized for math [multiplies]
- Embedded controller may not be a microcontroller *per se* but is used for special purpose control application
- Typical applications: temperature control, smart instrument, GPS, digital lock, cell phone, etc.



Examples

- Personal information products: Cell phone, pager, watch, pocket recorder, calculator
- Laptop components: mouse, keyboard, modem, fax card, sound card, battery charger
- Home appliances: door lock, alarm clock, thermostat, air conditioner, tv remote, hair dryer, VCR, small refrigerator, exercise equipment, washer/dryer, microwave oven
- Toys; video games, cars, dolls, etc.
- Cars are about 20-30% silicon today, mostly microcontrollers (\$4b/yr)
- Smart cards [credit cards plus]
- Usually anything with a keypad [simple calculators however have dedicated calculator chips]

Assignment 1: Due Next Thursday

- One page [no more] written assignment *to be handed in*.
- Find a microcontroller based device or product [i.e. Logitech optical computer mouse].
- Determine what *specific* microcontroller is used in the device [i.e. PIC 16F874].
- Explain the function or functions of the microcontroller in the device [i.e. converts pulses from rotary encoder to serial RS 232 communication protocol for transmission to computer input port].
- Be prepared to present in class.

Microcontroller Families

- Most manufacturers offer a wide range of devices for low end to higher end applications
- Microchip shipped its 1 billionth microcontroller in the fall of 1999, the 2 billionth in spring of 2002, and its 3 billionth in winter of 2004.



Microcontroller Manufacturers

- Analog Devices
- Atmel
- Dallas Semiconductor
- Freescale Semiconductor
- Hitachi Semiconductor
- Intel
- Microchip
- National Semiconductor
- Renesas
- STMicro
- Texas Instruments
- Zilog



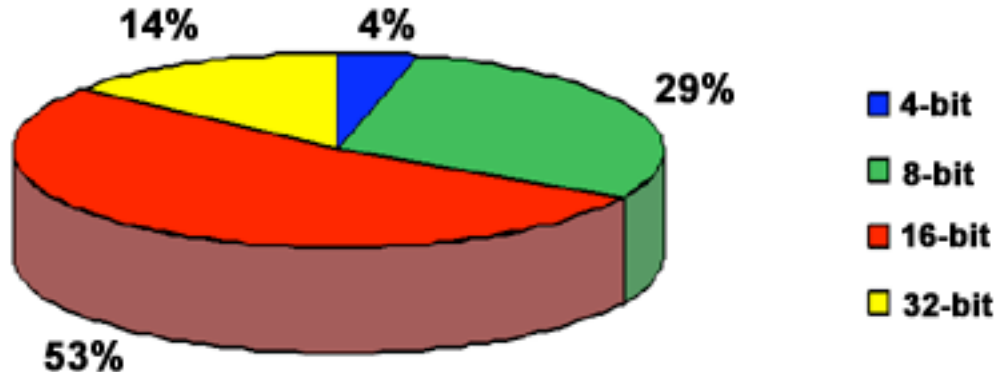
Microcontroller Market

- >40 suppliers, >50 architectures
- \$26 Billion market
- Shipments- > 16 Billion in 2000, 8 bit > 1/2 market
- Major Players: Microchip 16Fxx, Intel 8051, Motorola MC68HCXX, National COP800, SGS/Thomson ST62, Zilog Z86Cxx



Microcontrollers

- Processing power: 4 bit, 8 bit, 16 bit, 32 bit
- 2003 market share



- Specific features: communications, keyboard handling, signal processing, video processing

Embedded Controller- components

- ALU (arithmetic logic unit)
- RAM (Random Access Memory)
- EEPROM (Electrically Erasable Programmable Read Only Memory)
- I/O (input/output) - serial and parallel
- Timers [typically three designated 0,1,2]
- A/D converter
- Clock
- USART [univ. synchronous/asynch receiver and xmtr]
- Interrupt controller [[PIC16F874 diagram](#)]



16F87x Family: Features

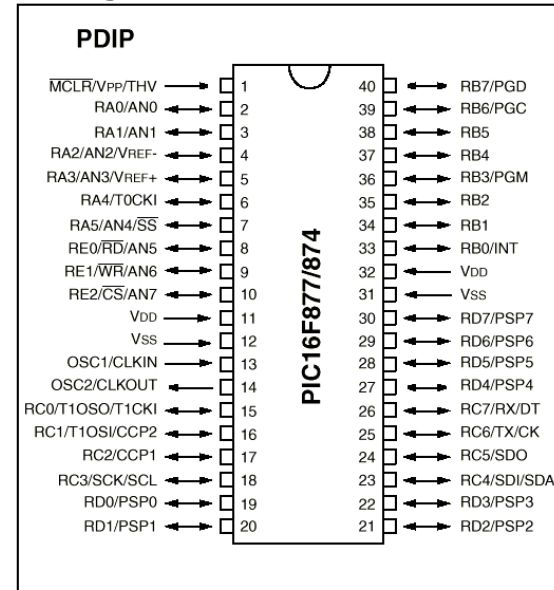
Devices Included in this Data Sheet:

- PIC16F873
- PIC16F876
- PIC16F874
- PIC16F877

Microcontroller Core Features:

- High-performance RISC CPU
- Only 35 single word instructions to learn
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC - 20 MHz clock input
DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory,
Up to 368 x 8 bytes of Data Memory (RAM)
Up to 256 x 8 bytes of EEPROM data memory
- Pinout compatible to the PIC16C73B/74B/76/77
- Interrupt capability (up to 14 sources)
- Eight level deep hardware stack
- Direct, indirect and relative addressing modes
- Power-on Reset (POR)
- Power-up Timer (PWRT) and
Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC
oscillator for reliable operation
- Programmable code-protection
- Power saving SLEEP mode
- Selectable oscillator options
- Low-power, high-speed CMOS FLASH/EEPROM
technology
- Fully static design
- In-Circuit Serial Programming™ (ICSP) via two
pins
- Single 5V In-Circuit Serial Programming capability
- In-Circuit Debugging via two pins
- Processor read/write access to program memory
- Wide operating voltage range: 2.0V to 5.5V
- High Sink/Source Current: 25 mA
- Commercial and Industrial temperature ranges
- Low-power consumption:
 - < 2 mA typical @ 5V, 4 MHz
 - 20 µA typical @ 3V, 32 kHz
 - < 1 µA typical standby current

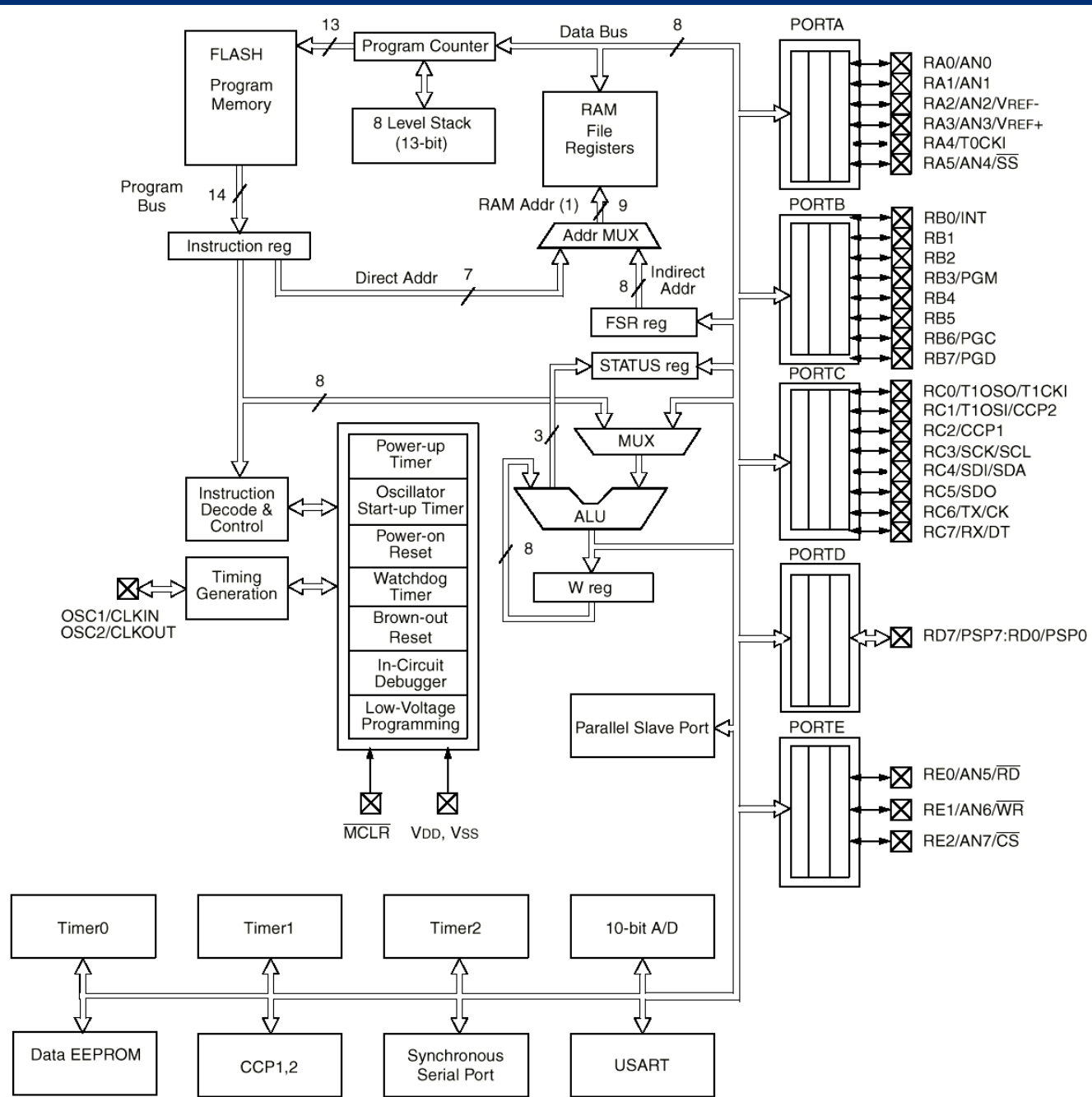
Pin Diagram



Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler,
can be incremented during sleep via external
crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period
register, prescaler and postscaler
- Two Capture, Compare, PWM modules
 - Capture is 16-bit, max. resolution is 12.5 ns
 - Compare is 16-bit, max. resolution is 200 ns
 - PWM max. resolution is 10-bit
- 10-bit multi-channel Analog-to-Digital converter
- Synchronous Serial Port (SSP) with SPI™ (Master
Mode) and I²C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver
Transmitter (USART/SCI) with 9-bit address
detection
- Parallel Slave Port (PSP) 8-bits wide, with
external RD, WR and CS controls (40/44-pin only)
- Brown-out detection circuitry for
Brown-out Reset (BOR)

16F874 Block Diagram



Note 1: Higher order bits are from the STATUS register.

TABLE 13-2: PIC16CXXX INSTRUCTION SET

16F87x
Instruction Set
[35]

Mnemonic, Operands	Description	Cycles	14-Bit Opcode				Status Affected	Notes	
			MSb	LSb					
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	1,2
CLRF	f	Clear f	1	00	0001	1fff	ffff	Z	2
CLRW	-	Clear W	1	00	0001	0xxx	xxxx	Z	
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	1,2
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2,3
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	1,2
MOVWF	f	Move W to f	1	00	0000	1fff	ffff		
NOP	-	No Operation	1	00	0000	0xx0	0000		
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		1,2
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3
LITERAL AND CONTROL OPERATIONS									
ADDLW	k	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z	
CALL	k	Call subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}, \overline{PD}$	
GOTO	k	Go to address	2	10	1kkk	kkkk	kkkk		
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLW	k	Move literal to W	1	11	00xx	kkkk	kkkk		
RETFIE	-	Return from interrupt	2	00	0000	0000	1001		
RETLW	k	Return with literal in W	2	11	01xx	kkkk	kkkk		
RETURN	-	Return from Subroutine	2	00	0000	0000	1000		
SLEEP	-	Go into standby mode	1	00	0000	0110	0011	$\overline{TO}, \overline{PD}$	
SUBLW	k	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z	

MPLAB IDE

- IDE = Integrated Development Environment
- MPLAB Editor
- MPLAB Assembler
- MPLAB ICD [in circuit debugger]
- MPLAB SIM [simulator]
- Programmer with ICD module



Blink.c

- Initial()
 - Initialize internal registers
- Blink()
 - Blink the LEDs
- LoopTime()
 - Wait for 10 milliseconds
- interrupt handler()
 - Check for interrupts from buttons



Summary

- Microprocessors and embedded controllers are a ubiquitous part of life today
- These devices come in a wide variety of configurations and designs
- Headhunters report that EEs familiar with μC , μP design are in the highest possible demand
- One pager due next Lecture !

More Information

- In the next session we will explore in more depth key features of μ Controllers
- Boards and ICDs in lab.
- All major Manufacturers have web sites

