MICROPROCESSORS B (17.384)

Spring 2011

Lecture Outline

Class # 01

January 25, 2011

Dohn Bowden

Today's Lecture

- Administrative
- General Course Overview
- Microcontroller Hardware and/or Interface
- Programming/Software
- Lab
- Homework

Course Admin

Administrative

- Admin for tonight …
  - Attendance/Introductions/Backgrounds
  - Syllabus
  - Textbook
  - 17.384 Web Site
  - Email List creation
  - Course Objectives

Attendance / Introductions / Backgrounds

- Attendance …
  - When called … please introduce yourself
    - Include the following
      - Microcontroller background
        - » When did you take 17.383 (Microprocessors A)?
      - Knowledge of "C" programming
      - Education
      - Work Experience
      - Other notable work/engineering/hobbies
      - Future Plans

My Background

- Education
- Work Experience
- Other notable work/engineering/hobbies
- Future Plans

Syllabus

- Syllabus …
  - Hard copies will be distributed
  - Electronic copy available on the class website
    - Web Address on syllabus
- Syllabus details …
  - New course …
    - Syllabus most likely will change as we get into the semester
  - Go over syllabus … next slide

Syllabus Review

Week	Date	Topics	Lab	Lab Report Due
1	01/25/11	PIC pin out, C programming, Watchdog Timer, Sleep		
2	02/01/11	General-purpose IO, LED/switch IO, FSM	1	
3	02/08/11	Lab	1 con't	
4	02/15/11	Interrupts, Timers, interrupt-driven IO	2	1
5	02/22/11	Lab	2 con't	
6	03/01/11	Asynchronous and Synchronous Serial IO (UART, I^2C, SPI)	3	2
7	03/08/11	**Examination 1**		
X	03/15/11	**No Class – Spring Break**		
8	03/22/11	Lab	3 con't	
9	03/29/11	Serial EEPROM operation, DAC, DC motor control, Servos, Stepper motor control	4	3
10	04/05/11	Lab	4 con't	
11	04/12/11	Advanced Hardware Topics	Project	4
12	04/19/11	**Examination 2**		
13	04/26/11	Work on Course Project	Project	
14	05/03/11	Final Exam/Course Project Brief and Demonstration	Demo	

Grading Policy

- Located at the bottom of syllabus
- Exam # 1 (20%) Exam #2 (20%)
- Laboratory … including lab reports (30%)
- Final Exam/Course Project (30%)

A	93-100	A-	90-92		
B+	87-89	B	83-86	B-	80-82
C+	77-79	C	73-76	C-	70-72
D+	67-69	D	60-66		
F	Below 60				

Class Hours

- Tuesdays evenings … 6 – 9 PM in BL-407
    - See next slide for access to BL-407
    - See syllabus for schedule of classes

- Please email me if you will not be in class …

- I am available for extra help ***Before*** / ***After*** class
    - If possible … please schedule in advance so I will ensure that I am available

- Labs – start to pick-up at approximately 8:45 PM

Access to Labs

- To gain access to BL-407
  - You will need to have an access cards
  - Access cards are given out on the South Campus at Access Services
    - Arrangements are done through the Continuing Education Office
  - I need your UMS# (will be on your Access Card) for room access

Textbooks

- Required Text …
  - Reese/Bruce/Jones, *Microcontrollers – From Assembly Language to C Using the PIC24 Family*, Course Technology/Cengage Learning, 2009
- *OPTIONAL TEXT* …
  - Stephen G. Kochan, *Programming in C,* Sams Publishing, 2005
    - An excellent book for those less familiar with C
- In addition … we will be using material that is available from the web
  - The course website will have a link to the material

Course Web Site

- The Course Web site Homepage is at:

  http://faculty.uml.edu/dbowden

- This website will contain the following:

  - Syllabus
  - Lab material
    - Labs procedures
    - Datasheets
  - Reference documents
    - Such as the textbook material
    - Links
  - Class lectures (will be placed on the web site *AFTER* the lecture)
  - Homework

Email Distribution List

- I will be creating a class email list
- Email me at …

  Dohn_Bowden@uml.edu

- This will ensure that your correct email address or addresses are included
- The email list will allow me to provide information to each of you

Chat Page

- ***NEW*** --- Any interest in having a chat page?
  - Not sure what is available …

Course Objectives

- What do you want to get out of this class?
- My goals for the course …

Course Evaluations

- How they are used

Questions?

General Course Overview

General Overview

- This is a "Hands-on" course …
    - The best way to learn is by doing!
- Similar to learning how to drive a car
    - Proficiency through experience
- Experience with the microcontroller by …
    - Applying the microcontroller to multiple hardware and software configurations

Course Expectations
Slide 1 of 2

- Continuation of programming and interfacing to the PIC microcontroller using a more advanced PIC
  - Programming the PIC microcontroller in both …
    - "C" and …
    - Assembly language
  - Interfacing the microcontroller with …
    - Commonly used electronic interfaces
      - Such as switches and Sensors
      - LEDs
      - LCDs

Course Expectations
Slide 2 of 2

- Continue to learn the how to utilize the microcontroller
  - Understanding the datasheet specifications
  - Differences between various microcontrollers

- Learn how to interface with devices
  - Understanding the electronics

- Course Project …
  - Applying the techniques and information covered during the course

Embedded Systems

- As stated last semester …
  - An ***embedded system*** is a combination of computer hardware and software, designed **to perform a dedicated function**
    - Examples … microwave oven, digital watch, video game player
- The design of an embedded system to perform a dedicated function is in direct contrast to that of the personal computer
  - A personal computer is **not designed to perform a specific function**, rather it is able to do many things

Embedded Developer

- The embedded developer needs to understand …
  - Hardware
  - Code
  - Peripheral interfaces

Embedded Language

- C programming language is the current language of choice
- Assembly Language

Getting to Know the Hardware

- Before writing software for an embedded system …
  - You must be familiar with the hardware on which it runs
- Understand …
  - General operation of the system
  - What the inputs are
  - What the outputs are
  - etc
- Initially you don't need all the details of the hardware, but it is helpful

Hardware Basics

- Understanding the hardware by reviewing …
  - The Schematics
    - Which give details of the hardware
  - Datasheets
    - Complete specifications for a specific component

The Processor

- Review the datasheets
- What internal functions does the processor have?
    - ADC?
    - Comparators?
    - Etc.
- What is connected to it?
- How does it communicate with those interfaces?
- Memory Mapping
- Initializing the processor

General Course Overview

- Typical Lecture/Class Structure
  - Microcontroller Hardware and/or Interface
  - Programming/Software
    - Programming Basics as we use them
    - "C" and/or Assembly Language Commands
    - PIC commands
  - Lab
    - Peer Review of Code (just prior to starting the lab for the night)
    - Overview
    - Lab Conduct
  - Homework

Microcontroller Hardware and / or Interfaces

Microprocessors A ...

A review ...

Microprocessors A
Summary

- Microcontrollers intended as a single chip solution
- Microprocessors require external support chips
  - Memory
  - Interface
- Microcontrollers have on-chip non-volatile memory for program storage …
  - Microprocessors do not

Microprocessors A
Summary

- Microcontrollers have more interface functions on-chip …
  - Serial interfaces
  - Analog-to-digital converters
  - Timers
  - Etc

Microprocessors A
Summary

- Microcontrollers do not have virtual memory support …
  - They cannot run Linux
- Microprocessors could run Linux
- General purpose microprocessors typically have higher performance over the microcontroller …
  - Clock speed
  - Data width
  - Instruction set
  - Cache

Microprocessors A
Summary

- The division between microprocessors and microcontrollers is ...
  - Becoming increasingly blurred!

Microprocessors A

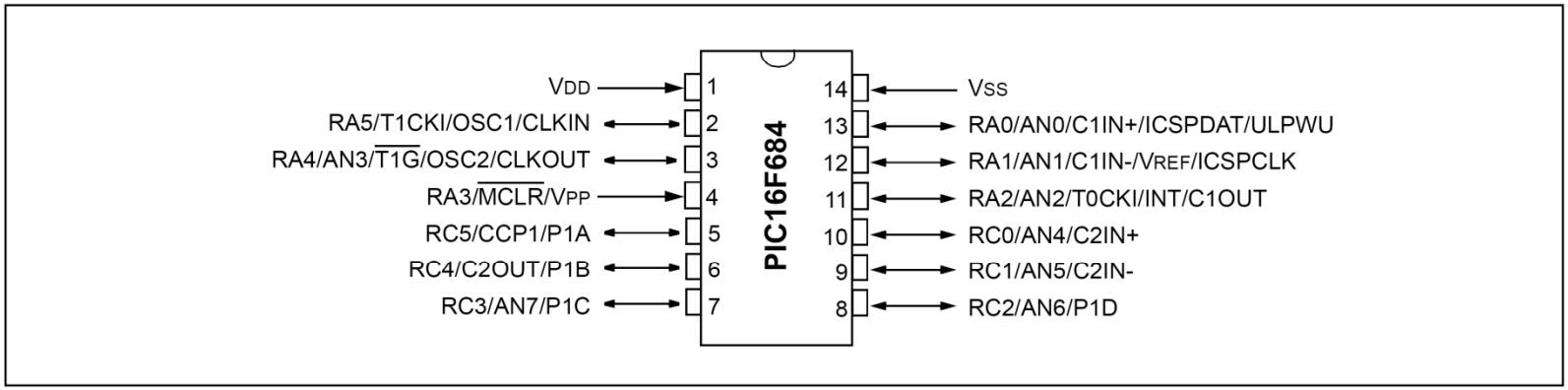
- You shall apply the material from Microprocessors A
  - On-chip peripherals/Hardware interface
    - PIC programming
  - C programming
  - Assembly Language
- … to this semester
  - New processor
  - Additional on-chip peripherals

PIC16F684 ...

PIC16F684

- We used the PIC16F684 during Microprocessors A (Fall 2010)
- The 684 gave us a good first look at microcontrollers
- However … there were some limitations
  - Program size
  - On-chip peripherals … limited number
  - Instructions (35)

PIC16F684


PIC16F684
VDD
1
RA5/T1CKI/OSC1/CLKIN
2
RA4/AN3/T1G/OSC2/CLKOUT
3
RA3/MCLR/VPP
4
RC5/CCP1/P1A
5
RC4/C2OUT/P1B
6
RC3/AN7/P1C
7
14
VSS
13
RA0/AN0/C1IN+/ICSPDAT/ULPWU
12
RA1/AN1/C1IN-/VREF/ICSPCLK
11
RA2/AN2/T0CKI/INT/C1OUT
10
RC0/AN4/C2IN+
9
RC1/AN5/C2IN-
8
RC2/AN6/P1D

PIC16F684

Device	Program Memory	Data Memory		I/O	10-bit A/D (ch)	Comparators	Timers 8/16-bit
	Flash (words)	SRAM (bytes)	EEPROM (bytes)				
PIC16F684	2048	128	256	12	8	2	2/1

Microprocessors B ...

This Semester …

- We shall look at devices that contains …
  - Larger memory for larger programs
  - More on-chip peripherals
  - Greater Instructions set (71)
- We shall use the PIC24HJ32GP202 microcontroller

Introduction to the PIC24 Microcontroller Family ...

PIC24 Microcontroller Family Overview

- PIC24 …
  - High speed
  - 3.0 to 3.6 volt external supply
  - 40 million instructions per second (MIPS) maximum

- All PIC24 family of microcontrollers use the same instruction set architecture (ISA)

  - Subset of the dsPIC30/dsPIC33 ISA …
    - dsPIC30/dsPIC33 is another 16-bit family from Microchip

- Instruction width … 24 bits

The PIC24HJ32GP202

- The PIC24HJ32GP202
  - PIC … Microchip microcontroller
  - 24 … Family of chips
  - H … High Speed (F is the lower speed variant)
  - J …
  - 32 … Program memory size in kiliobinary bytes
  - GP … General Purpose
  - 202 …

PIC24HJ32GP202 Features

• Architecture	16-bit
• CPU Speed (MIPS)	40
• Memory Type	Flash
• Program Memory (KB)	32
• RAM Bytes	2,048
• Temperature Range C	-40 to 140
• Operating Voltage Range (V)	3 to 3.6
• I/O Pins	21
• Pin Count	28
• System Management Features	PBOR
• Internal Oscillator	7.37 MHz, 512 kHz
• nanoWatt Features	Fast Wake/Fast Control
• Digital Communication Peripherals	1-UART, 1-SPI, 1-I2C

http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en530328

PIC24HJ32GP202 Features

• Analog Peripherals	1-A/D 10x12-bit @ 500(ksps)
• Comparators	0
• CAN (#, type)	0, None
• Capture/Compare/PWM Peripherals	4/2
• 16-bit PWM resolutions	16
• Timers	3 x 16-bit 1 x 32-bit
• Parallel Port	GPIO
• Hardware RTCC	No
• DMA	0
• Cap Touch Channels	10

http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en530328

PIC24 Microcontroller Family Overview

- Instruction width …
  - 24 bits

- On-chip program memory (non-volatile, electrically erasable) …
  - PIC24HJ32GP202 has 32K bytes/11264 instructions
  - architecture supports 24Mbytes/4M instructions

- On-chip Random Access Memory (RAM) …
  - PIC24HJ32GP202 has 2048 bytes
  - architecture supports up 65536 bytes

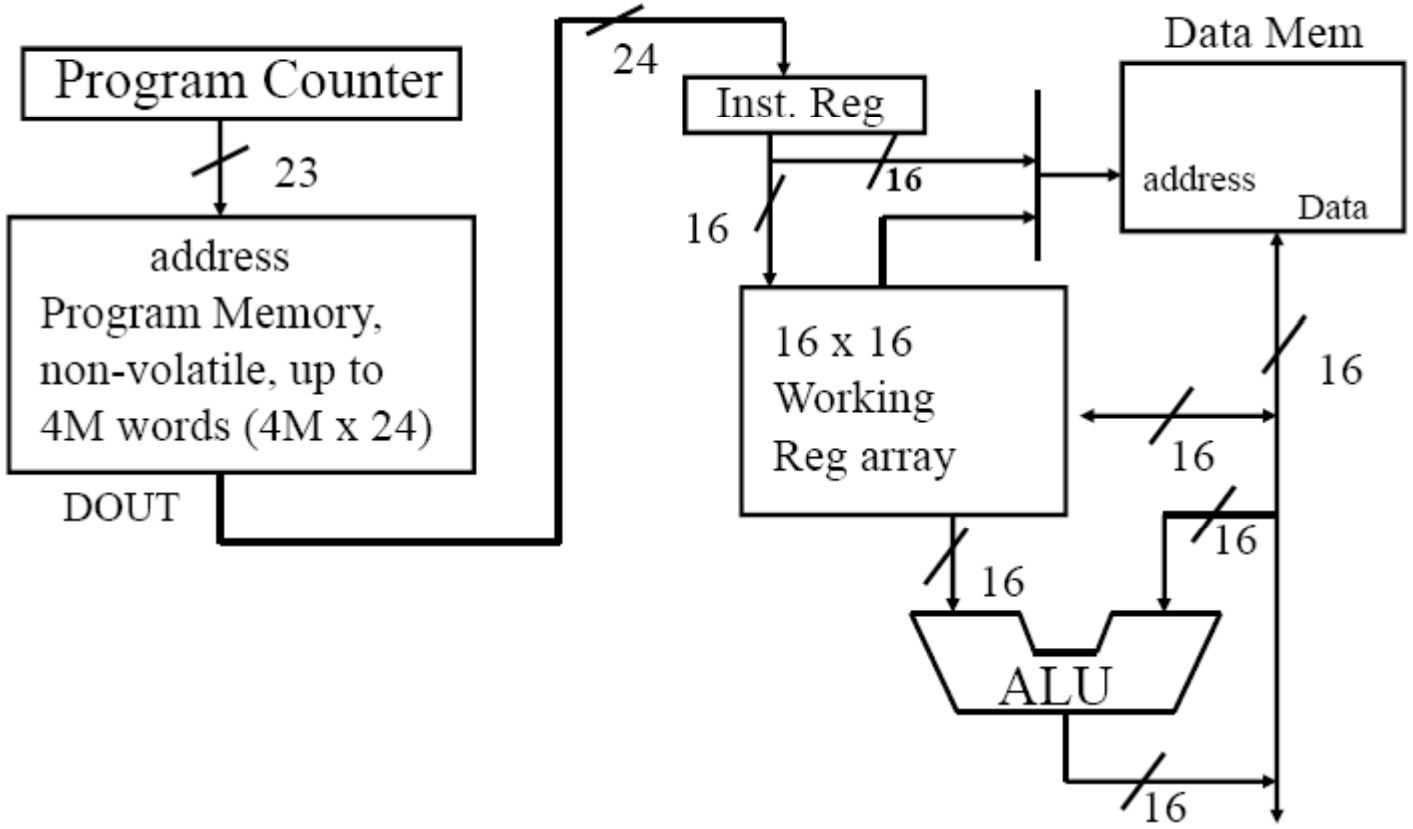
PIC24 Microcontroller Family Overview

- Clock speed …
  - DC to 80 MHz
- Architecture …
  - General purpose registers
  - 71 instructions not including addressing mode variants
- On-chip modules …
  - Async serial IO, I2C, SPI, A/D, three 16-bit timers, one 8-bit timer, comparator

PIC24

- Internal data paths of the CPU is …
  - 16 bits
- Therefore … the PIC24 is referred to as a 16-bit microcontroller
- This means …
  - Natural size of the computation is 16 bits
    - 16 bit or 8 bit data can take 1 instruction cycle
    - Larger than 16 bit requires multiple instructions

PIC24 Core (Simplified Block Diagram)



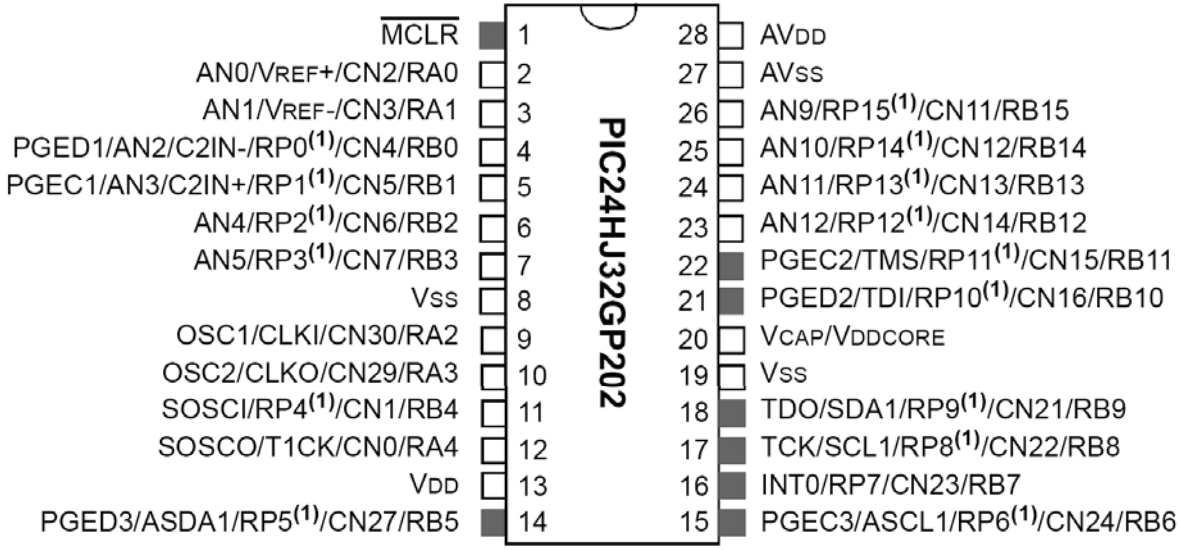
PIC24 Core

- Instruction word …
  - 24-bit machine code
- Two instruction types require …
  - Two instruction words (6 bytes) to encode

PIC24HJ32GP202

28-Pin SDIP, SOIC

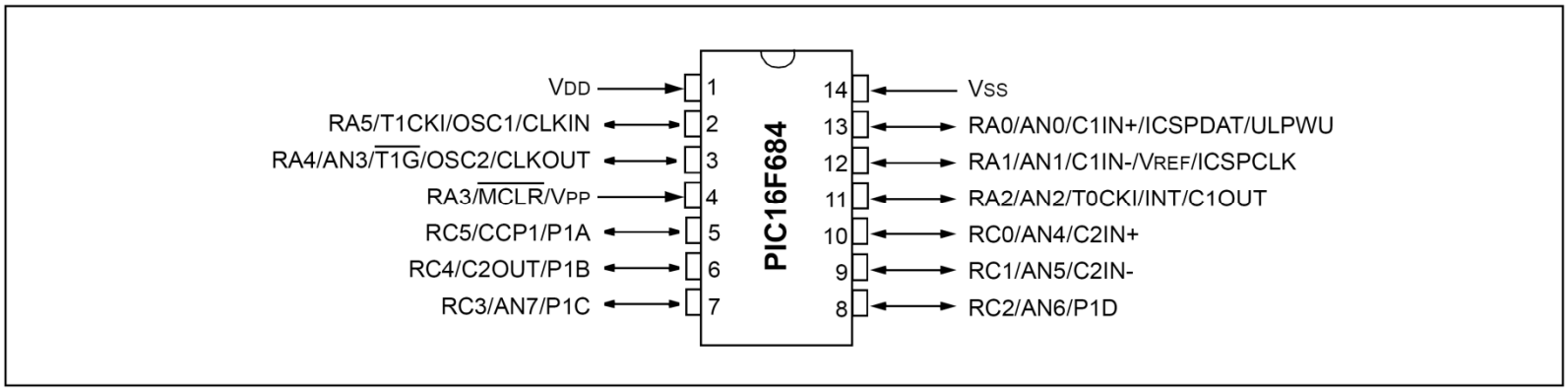
■ = Pins are up to 5V tolerant



PIC24HJ32GP202/204 AND PIC24HJ16GP304 CONTROLLER FAMILIES

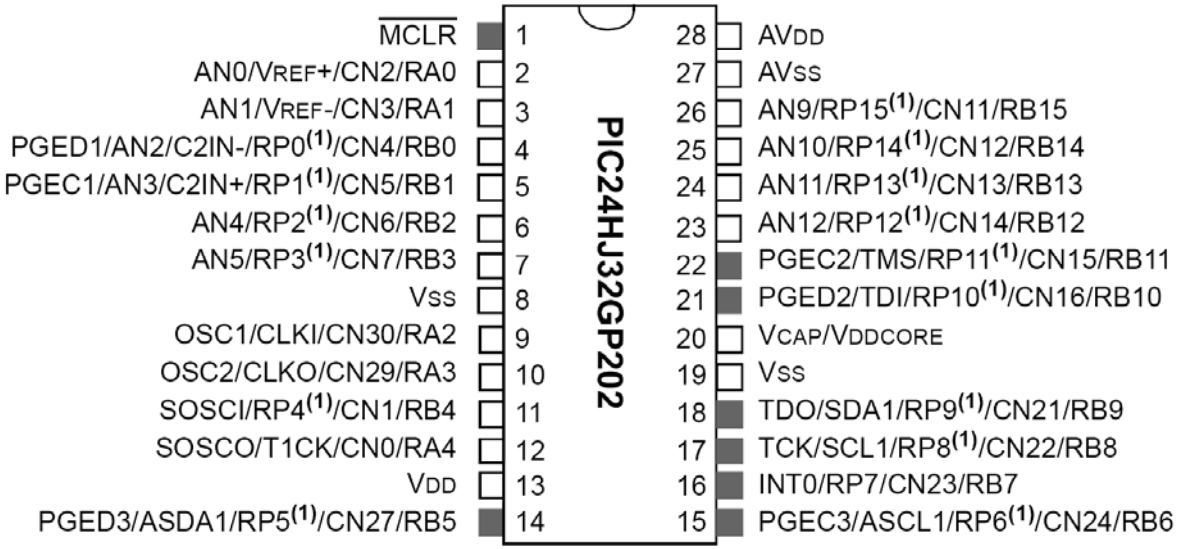
Device	Pins	Program Flash Memory (Kbyte)	RAM	Remappable Peripherals							10-Bit/12-Bit ADC	I^2C™	I/O Pins (Max)	Packages
				Remappable Pins	16-bit Timer	Input Capture	Output Compare Std. PWM	UART	External Interrupts(2)	SPI				
PIC24HJ32GP202	28	32	2	16	3(1)	4	2	1	3	1	1 ADC, 10 ch	1	21	SDIP SOIC QFN-S
PIC24HJ32GP204	44	32	2	26	3(1)	4	2	1	3	1	1 ADC, 13 ch	1	35	QFN TQFP
PIC24HJ16GP304	44	16	2	26	3(1)	4	2	1	3	1	1 ADC, 13 ch	1	35	QFN TQFP

PIC16F684/PIC24 Comparison



28-Pin SDIP, SOIC

■ = Pins are up to 5V tolerant



PIC16F684/PIC24 Comparison

Device	Program Memory	Data Memory		I/O	10-bit A/D (ch)	Comparators	Timers 8/16-bit
	Flash (words)	SRAM (bytes)	EEPROM (bytes)				
PIC16F684	2048	128	256	12	8	2	2/1

Device	Pins	Program Flash Memory (Kbyte)	RAM	Remappable Peripherals							10-Bit/12-Bit ADC	I^2C^{TM}	I/O Pins (Max)	Packages
				Remappable Pins	16-bit Timer	Input Capture	Output Compare Std. PWM	UART	External Interrupts(2)	SPI				
PIC24HJ32GP202	28	32	2	16	3(1)	4	2	1	3	1	1 ADC, 10 ch	1	21	SDIP SOIC QFN-S
PIC24HJ32GP204	44	32	2	26	3(1)	4	2	1	3	1	1 ADC, 13 ch	1	35	QFN TQFP
PIC24HJ16GP304	44	16	2	26	3(1)	4	2	1	3	1	1 ADC, 13 ch	1	35	QFN TQFP

Memory ...

PIC Memory

- Memory on the PIC24 is split into two types …
  - Program Memory … and …
  - Data Memory
- Separate memories arrangement …
  - Is the Harvard architecture

Program Memory

- Instructions are stored in …
  - Program memory …
    - Which is non-volatile … contents are retained when power is lost
- A PIC24 instruction is 24 bits wide (3 bytes) …
  - PIC24HJ32GP202 program memory supports 11264 instructions
    - The PIC24 architecture can support up to 4M instructions

Program Memory

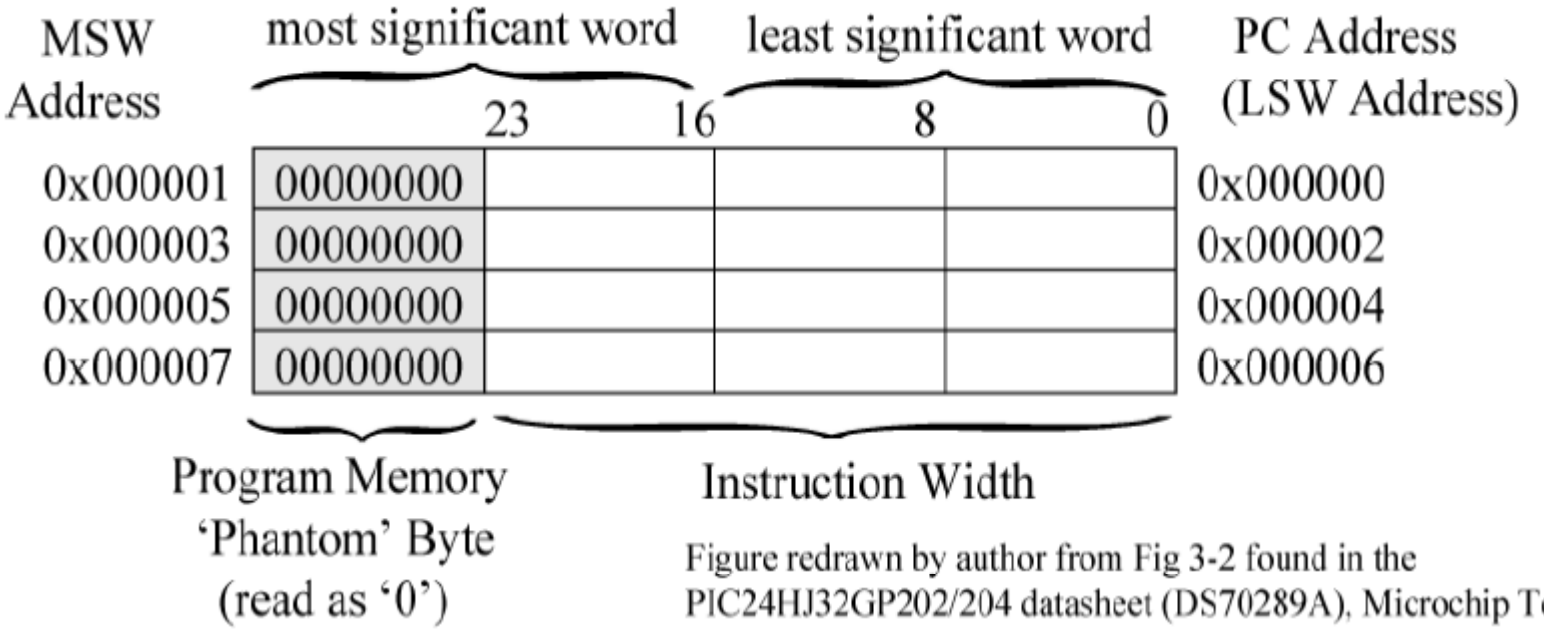


Figure redrawn by author from Fig 3-2 found in the PIC24HJ32GP202/204 datasheet (DS70289A), Microchip Technology Inc.

PC is 23 bits wide, but instructions start on even word boundaries (the PC least significant bit is always 0), so the PC can address 4 Mi instructions.

- Locations 0x000000-0x0001FF reserved
- User program starts at location 0x000200

Data Memory

- PIC24 data is stored in data memory …
  - Also known as the file registers …
    - And is a maximum size of 65536 x 8
  - Data memory is volatile … contents are lost when power is lost

Data Memory Organization

Data memory for PIC24HJ32GP202

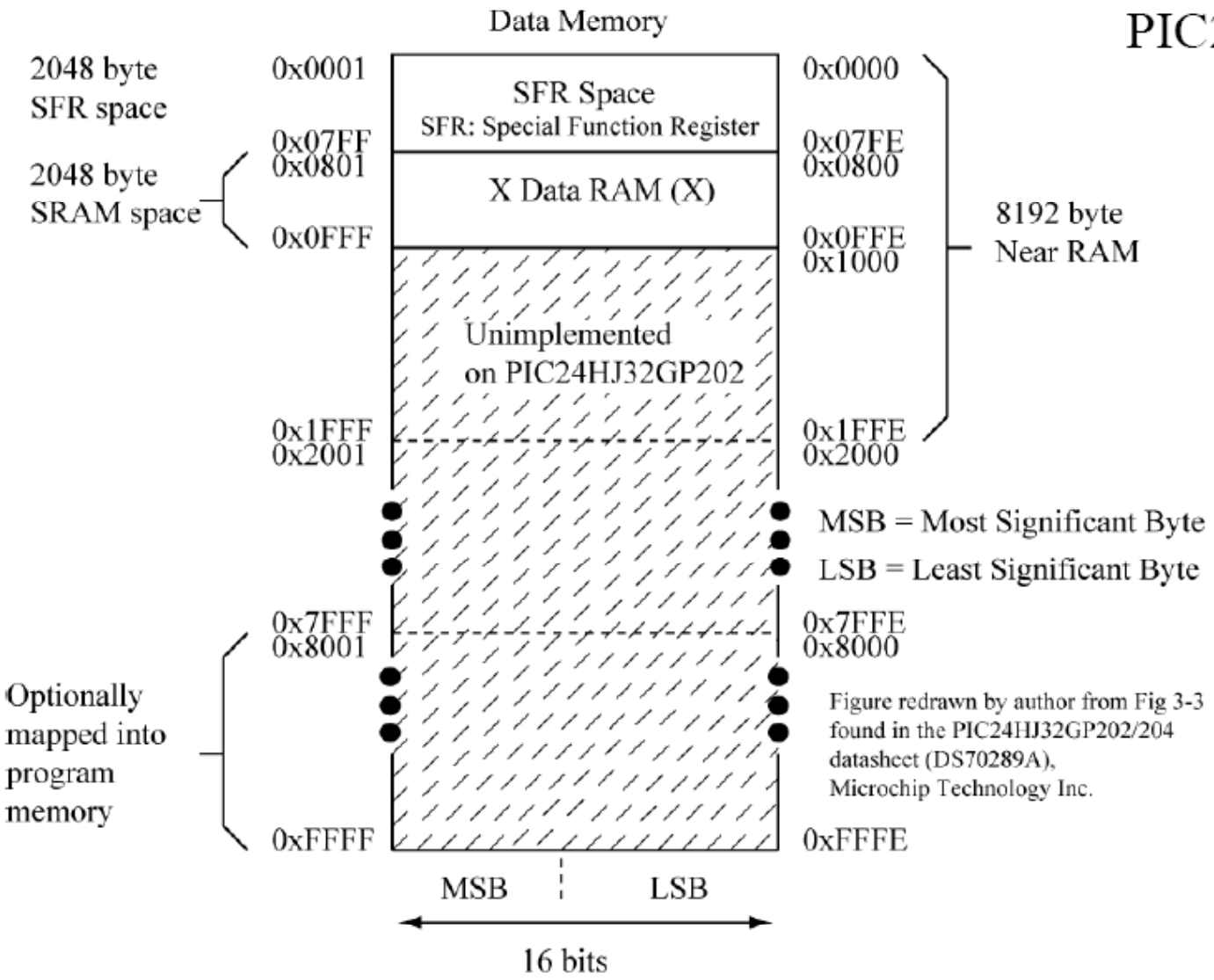


Figure redrawn by author from Fig 3-3 found in the PIC24HJ32GP202/204 datasheet (DS70289A), Microchip Technology Inc.

Special Function Registers (SFR) …

Special Function Registers(SFR)

- Addressed like normal data memory locations … but …
  - Have specified functionality tied to hardware subsystems in the processor
  - We typically refer to SFRs by name …
    - W0, T3CON, STATUS, etc … instead of by address
- SFRs are used as control registers … and … data registers for processor subsystems …
  - Like the serial interface, or the analog-to-digital converter)
  - We will cover their use and names as we need to

Special Function Registers(SFR)

- SFRs live in the address range 0x0000 to 0x07FE in data memory …
  - See the datasheet for a complete list of SFRs
- Other locations in data memory that are not SFRs can be used for …
  - Storage of temporary data
    - They are not used by the processor subsystems
    - These are sometimes referred to as GPRs (general purpose registers)
    - MPLAB refers to these locations as file registers

Data ...

8-bit, 16-bit, 32-bit Data

- We will deal with data that is (in size) …
  - 8 bits
  - 16 bits (2 bytes) … and …
  - 32 bits (4 bytes)

- Initially we will use only 8 bit and 16 bit

8-bit, 16-bit, 32-bit Data

Size	Unsigned Range
8-bits	0 to 2^8-1 (0 to 255, 0 to 0xFF)
16-bit	0 to 2^{16}-1 (0 to 65536, 0 to 0xFFFF)
32-bit	0 to 2^{32}-1 (0 to 4,294,967,295), 0 to 0xFFFFFFFF)

- The lower 8 bits of a 16-bit value or of a 32-bit value is known as …
  - The Least Significant Byte (LSB)
- The upper 8 bits of a 32-bit value is known as …
  - The Most Significant Byte (MSB)

Storing Multi-byte Values in Memory

- 16-bit and 32-bit values are stored in memory from …
  - Least significant byte to most significant byte …
    - In increasing memory locations (little endian order)

Storing Multi-byte Values in Memory

Assume the 16-bit value 0x8B1A stored at location 0x1000

Assume the 32-bit value 0xF19025AC stored at location 0x1002

Location	Contents	
0x1000	0x1A	← LSB
0x1001	0x8B	
0x1002	0xAC	← LSB
0x1003	0x25	
0x1004	0x90	
0x1005	0xF1	

Memory shown as 8 bits wide

Location	Contents	
0x1000	0x8B1A	← LSB
0x1002	0x25AC	← LSB
0x1004	0xF190	
0x1006	?????	
0x1008	?????	
0x1006	?????	

Memory shown as 16 bits wide

The LSB of a 16-bit or 32-bit value must begin at an even address (be *word aligned*).

Clock Cycles
vs
Instruction Cycles ...

Clock Cycles vs. Instruction Cycles

The clock signal used by a PIC24 μC to control instruction execution can be generated by an off-chip oscillator or crystal/capacitor network, or by using the internal RC oscillator within the PIC24 μC.

For the PIC24H family, the maximum clock frequency is 80 MHz.

An **instruction cycle** is **two clock** cycles. ← Important!!!!!!

A PIC24 instruction takes 1 or 2 **instruction** cycles, depending on the instruction (see Table 19-2, PIC24HJ32GP202 data sheet). If an instruction causes the program counter to change, that instruction takes 2 instruction cycles.

An add instruction takes 1 instruction cycle. How much time is this if the clock frequency is 80 MHz (1 MHz = 1.0e6 = 1,000,000 Hz)?

1/frequency = period, 1/80 MHz = 12.5 ns (1 ns = 1.0e-9 s)

1 Add instruction @ 80 MHz takes 2 clocks * 12.5 ns = 25 ns (or 0.025 us).

By comparison, an Intel Pentium add instruction @ 3 GHz takes 0.33 ns (330 ps). An Intel Pentium could emulate a PIC24HJ32GP202 faster than a PIC24HJ32GP202 can execute! But you can't put a Pentium in a toaster, or buy one from Digi-key for $5.00.

Units ...

Review: Units

In this class, units are always used for physical qualities:

Time	Frequency
milliseconds (ms = 10^{-3} s)	kilohertz (kHz = 10^{3} Hz)
microseconds (μs = 10^{-6} s)	megahertz (MHz = 10^{6} Hz)
nanoseconds (ns = 10^{-9} s)	gigahertz (GHz = 10^{9} Hz)

When a time/frequency/voltage/current quantity is asked for, I will always ask from some units. Values for these quantities in datasheets ALWAYS are given in units.

For a frequency of 1.25 kHz, what is the period in μs?

period = 1/f = 1/(1.25 e3) = 8.0 e –4 seconds

Unit conversion= 8.0e-4 (s) * (1e6 μs)/1.0 (s) = 8.0e2 μs = **800 μs**

PIC Family ...

PIC24H Family

- Microchip has an extensive line of PICmicro®microcontrollers, with the PIC24 family introduced in 2005
- The PIC16 and PIC18 are older versions of the PICmicro®family …
  - There have been several previous generations
- Do not assume that because something is done one way in the PIC24 … that it is the most efficient method for accomplishing that action
- The datasheet for the PIC24HJ32GP202 is found on the class web site

Chapter 8 ...

C and Embedded Systems

- A microcontroller-based system used in a device (i.e, a car engine) performing control and monitoring functions is …
  - Referred to as an **embedded system**
    - The embedded system is invisible to the user
    - The user only indirectly interacts with the embedded system by using the device that contains the microcontroller

C and Embedded Systems

- Why are programs written in C?
  - Portable … code can be retargeted to different processors
  - Clarity … C is easier to understand than assembly
  - Compilers produce code that is close to manually-tweaked assembly language in both code size and performance

So Why Learn Assembly Language?

- The way that C is written can impact assembly language size and performance …
  - i.e., if the **uint32** data type is used where **uint8** would suffice, both …
    - Performance and code size will suffer

So Why Learn Assembly Language?

- Learning the assembly language for the architecture of the target microcontroller provides …

  – Performance and code size clues for compiled C
    - Does the uP have support for multiply/divide?
    - Can it shift only one position each shift or multiple positions? (i.e, does it have a *barrel shifter?)*
    - How much internal RAM does the µP have?
    - Does the µP have floating point support?

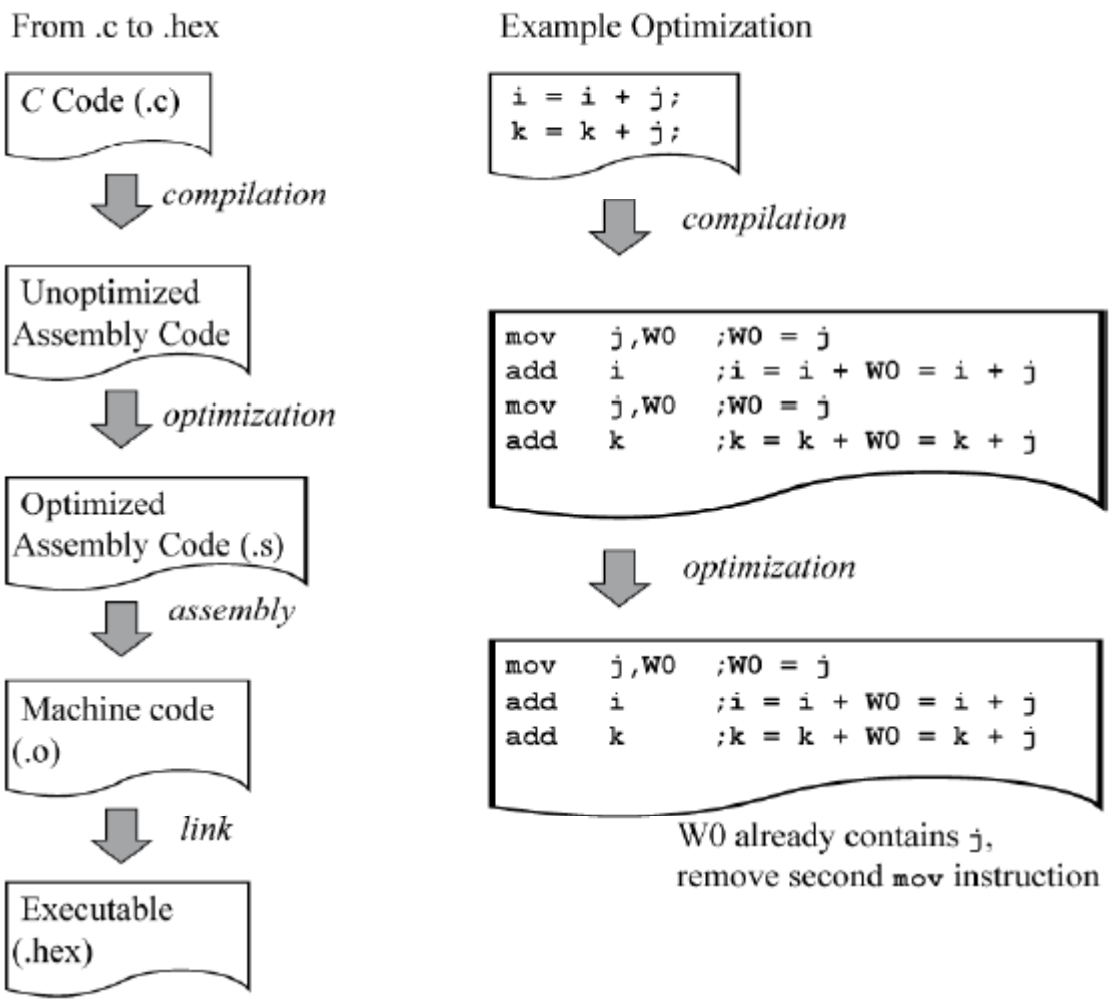
- Sometimes have to write assembly code for performance reasons

C Compilation ...

C Compilation

- A compiler is defined as …
  - A program that translates statements in a high-level language to assembly language
- The next slides shows a conceptual view of the steps performed in transforming a C program into machine code
- Machine code is what is actually programmed into the microcontroller

C Compilation



MPLAB PIC24 *C Compiler*

- We will be using C for our labs …
  - At times we may need to use assembly language
- Will use the MPLAB PIC24 C Compiler from Microchip
  - Excellent compiler, based on GNU C … which generates very good code
- Use the MPLAB example projects that come with the ZIP archive associated with the textbook

Special Function Registers ...

Special Function Registers

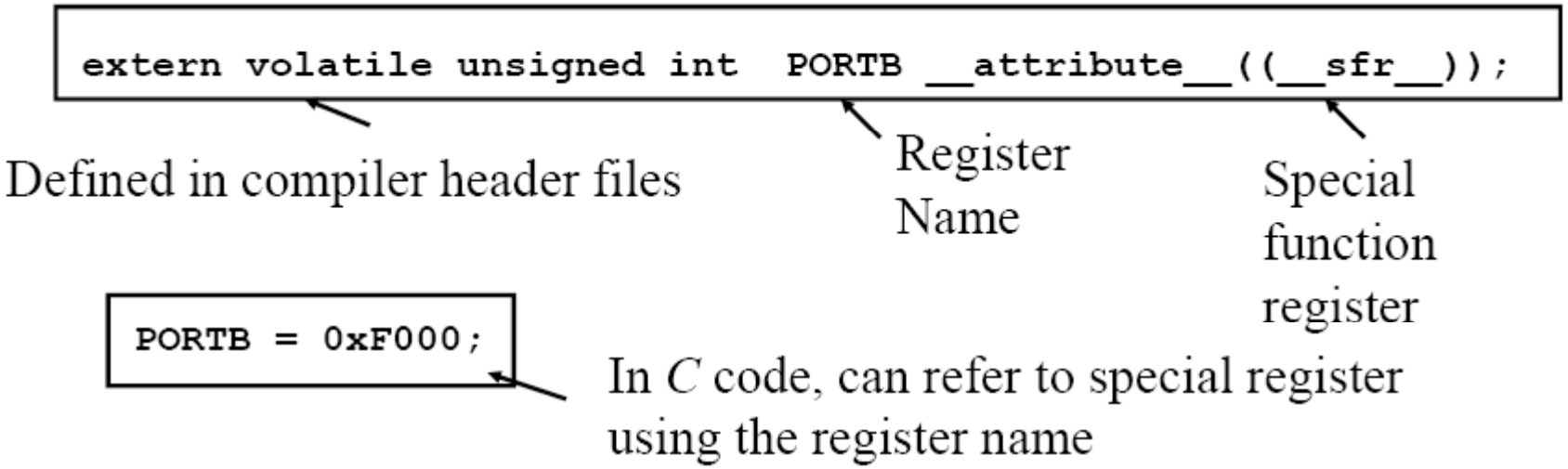
- Must include the following statement at top of a C file

```
#include "pic24.h"
```

  - This will include the all of the header files for the support libraries

Special Function Registers

- Special Function registers can be accessed like variables …



Referring to Bits within Special Registers

- The compiler include file also has definitions for individual bits within special function registers
  - Can use these to access individual bits and bit fields

```
PORTBbits.RB5 = 1;    //set bit 5 of PORTB
PORTBbits.RB2 = 0;    //clear bit 2 of PORTB

 if (PORTBbits.RB0) {
  //execute if-body if LSb of PORTB is '1'
....
}

OSCCONbits.NOSC = 2;     //bit field in OSSCON register
```

Referring to Bits within Special Registers

- Using *registername.bitname* requires you to remember both the register name and the bitname

- For bitnames that are UNIQUE, can use just *_bitname*

```
_RB5 = 1;    //set bit 5 of PORTB
_RB2 = 0;    //clear bit 2 of PORTB

 if (_RB0) {
  //execute if-body if LSb of PORTB is '1'
....
}

_NOSC = 2;     //bit field in OSSCON register
```

Breadboard Schematic ...

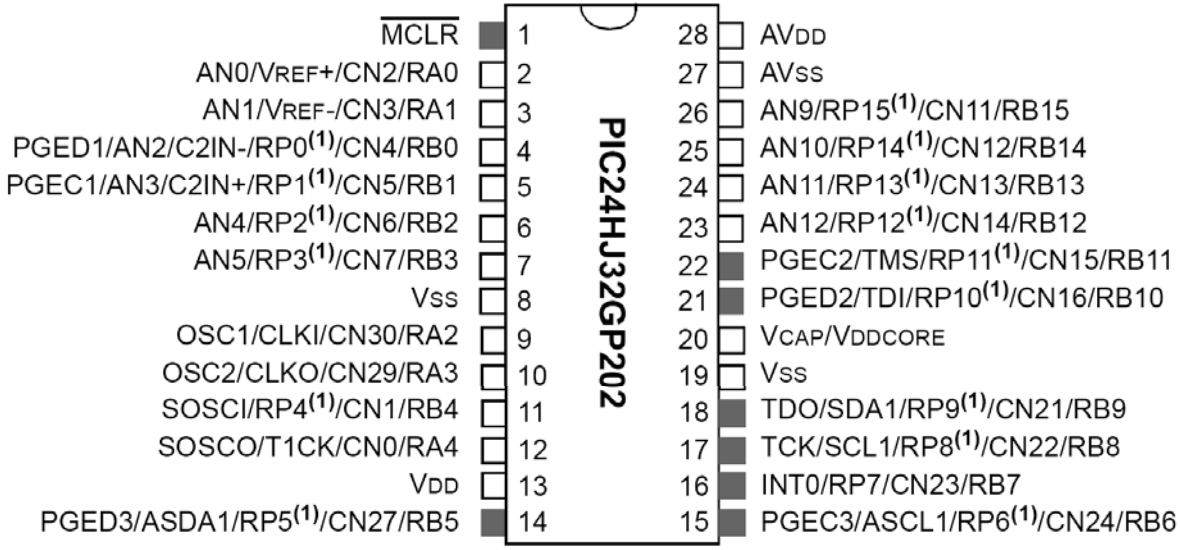
Labs

- As discussed … we will be using the …
  - PIC24HJ32GP202 microcontroller (28-pin DIP)
- Recall from earlier in the lecture …
  - Most pins have multiple functions
- Pin functions are controlled via special registers in the PIC

PIC24HJ32GP202

28-Pin SDIP, SOIC

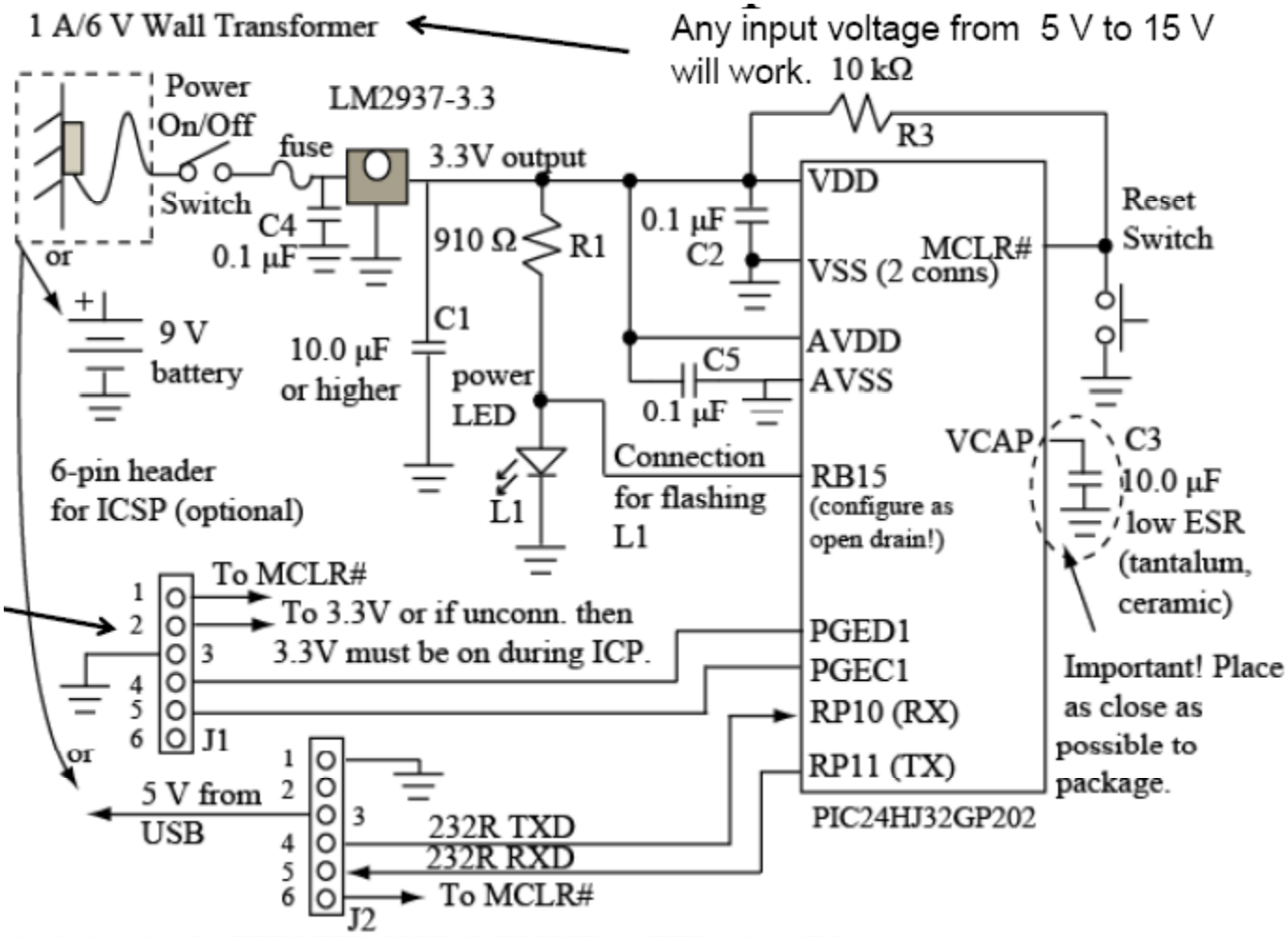
■ = Pins are up to 5V tolerant



Building the Circuits

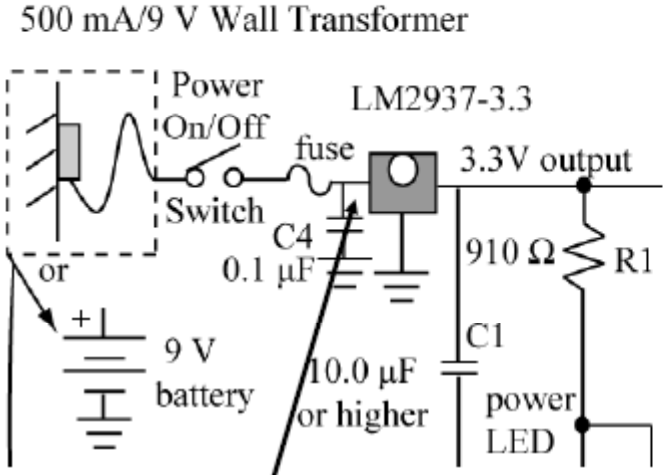
- We shall add to our circuits as we need to
  - This will allow ease in troubleshooting
  - Reduce time wiring
  - Reuse what we have already built

Initial Schematic



6-pin header for FTDI TTL-232R-3.3V USB-to-TTL cable (PC serial communication link)

Powering the PIC24 µC



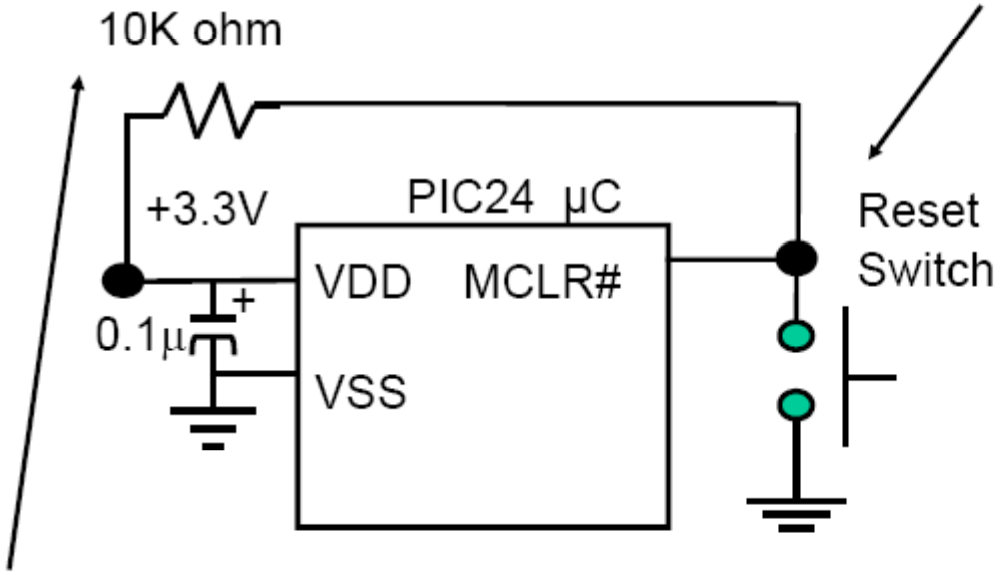
The POWER LED provides a visual indication that power is on.

A Wall transformer provides 15 to 6V DC unregulated (unregulated means that voltage can vary significantly depending on current being drawn). The particular wall Xfmr in the parts kit provides 6V with a max current of 1000 mA.

The LM2937-3.3 voltage regulator provides a regulated +3.3V. Voltage will stay stable up to maximum current rating of device.

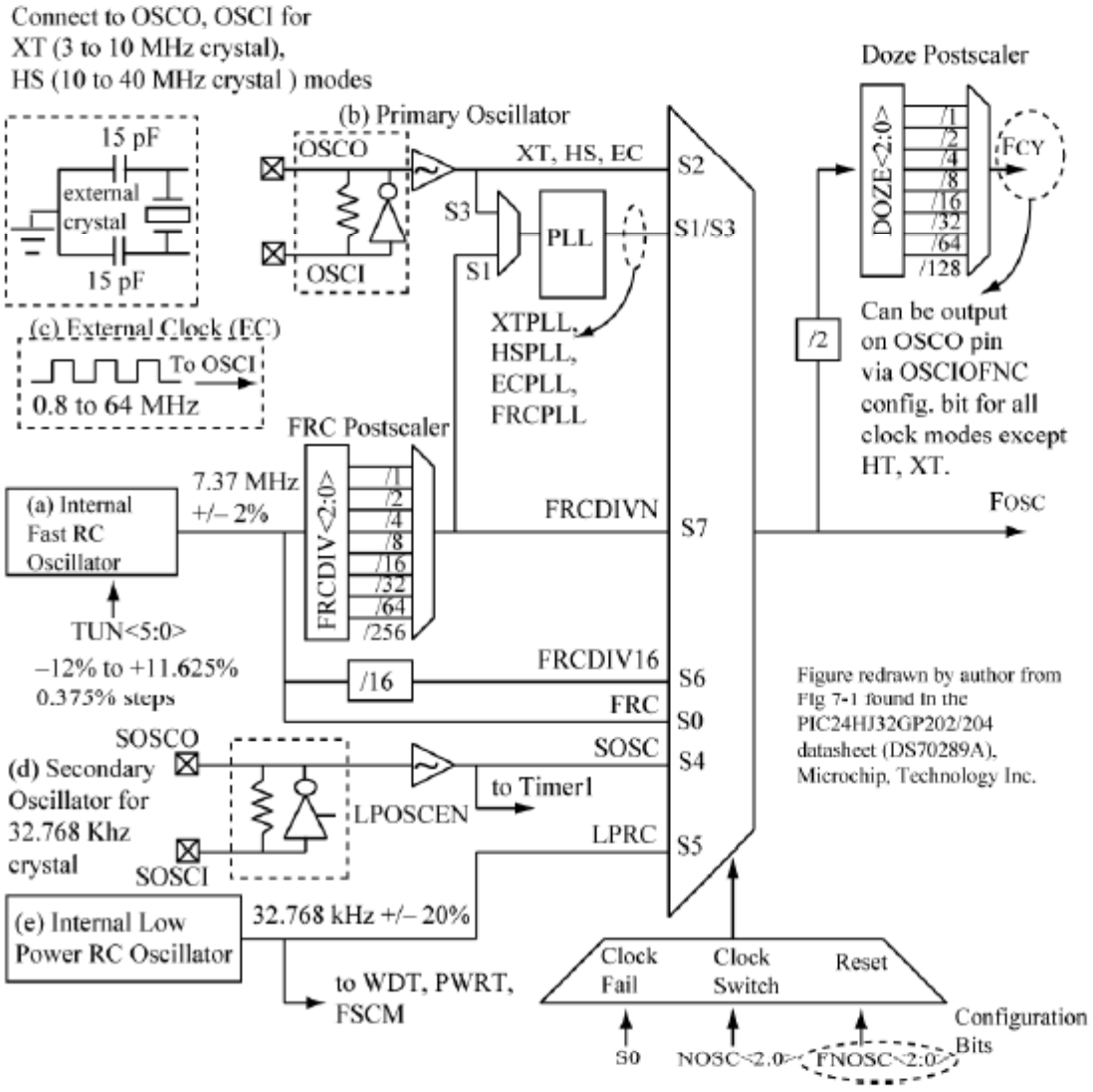
With writing on device visible, input pin (+9 v) is left side, middle is ground, right pin is +3.3V regulated output voltage.

Reset



10K resistor used to limit current when reset button is pressed.

When reset button is pressed, the MCLR# pin is brought to ground. This causes the PIC program counter to be reset to 0, so next instruction fetched will be from location 0. All µCs have a reset line in order to force the µC to a known state.



The Clock

The PIC24 μC has many options for generating a clock; can use an external crystal or internal oscillator.

We will use the internal clock.

Configuration Bits ...

Configuration Bits

- ***Configuration bits*** are …
  - Stored at special locations in program memory to control various processor options
  - Configuration bits are only read at power up

- Processor options controlled by configuration bits relate to …
  - Oscillator options
  - Watchdog timer operation
  - RESET operation
  - Interrupts
  - Code protection
  - Etc.

Configuration Bits

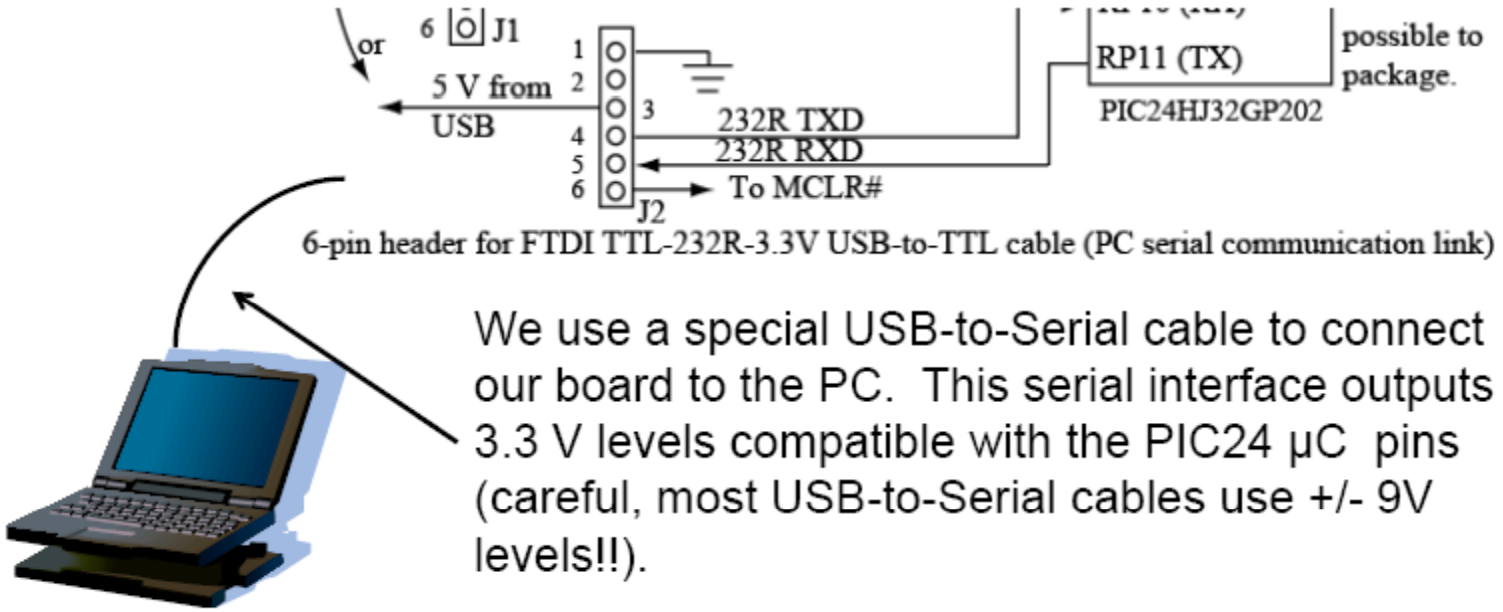
- The file ***pic24_config.c*** file …
  - Included by the sample programs used in the labs
    - Specifies configuration bits used for all lab experiments
  - C:\Program Files\Microchip\mplabc30\v3.25\support\PIC24H\h\p24hj32gp202.h
- We will discuss the meaning of the configuration bit options as it is necessary

The PC Serial Interface

- Later in the semester, the serial interface will be used for …
  - ASCII input/output to PIC24
  - Downloading new programs via Bully Serial Bootloader
    - winbootldr.exe

PC Serial Interface ...

The PC Serial Interface



In-circuit Serial Programming ...

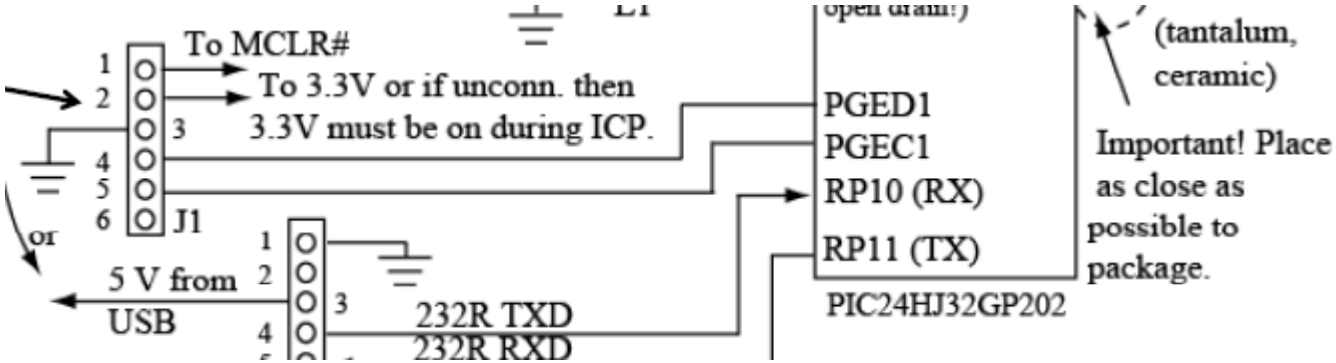
In-circuit Serial Programming

- We shall be programming the PIC24 with the …
  - PICkit 2

  - The PICkit 2 will connect to J1 … a 6 pin header

In-circuit Serial Programming

- J1 … used to connect the PICkit 2



PICkit 2

- If you desire …
  - PICkit 2's are available for purchase in the bookstore
  - Jumper kits are also available for purchase in the bookstore

Our First Program ...

First Program

- Like Microprocessors A …
  - We will be flashing an LED
    - This will provide us …
      - A first look at the new processor
      - New C compiler
      - New programmer
      - Test of our circuits

ledflash_nomacros.c

- ledflash_nomacros.c … shown on the next slide …
  - Flashes the L1 LED connected to pin RB15
  - It does not use any macros for the delay circuit

ledflash_nomacros.c

```
#include "pic24_all.h"
```

Includes several header files, discussed later in this chapter.

```
/**
A simple program that flashes the Power LED.
*/

//a naive software delay function
void a_delay(void){
  uint16 u16_i,u16_k;
  // change count values to alter delay
  for (u16_k = 1800; --u16_k;) {
    for(u16_i = 1200 ; --u16_i ;);
  }
}

int main(void) {
  configClock();    //clock configuration
  /********** PIO config **********/
  _ODCB15 = 1;          //enable open drain
  _TRISB15 = 0;         //Config RB15 as output
  _LATB15 = 0;          //RB15 initially low
  while (1) {           //infinite while loop
   a_delay();          //call delay function
   _LATB15 = !_LATB15; //Toggle LED attached to RB15
  } // end while (1)
}
```

A subroutine for a software delay. Change `u16_i`, `u16_k` initial values to change delay.

Infinite loop that blinks the LED. Only exit is through MCLR# reset or power cycle.

ledflash.c

- ledflash.c … shown on the next slide …
  - Flashes the L1 LED connected to pin RB15
    - Same as previous C program
  - It ***USES*** macros for the delay circuit

ledflash.c

```
#include "pic24_all.h"

/**
A simple program that
flashes an LED.
*/

#define CONFIG_LED1()  CONFIG_RB15_AS_DIG_OD_OUTPUT()

#define LED1    _LATB15

int main(void) {

  configClock();     //clock configuration
  /********** PIO config **********/
  CONFIG_LED1();    //config PIO for LED1
  LED1 = 0;

  while (1) {
    DELAY_MS(250);  //delay
    LED1 = !LED1;   // Toggle LED
  } // end while (1)
}
```

Defined in device-specific header file in *include\devices* directory in the book source distribution. Macro `CONFIG_RB15_AS_DIG_OD_OUTPUT()` configures RB15 as an open drain output and contains the statements `_TRISB15=0, _ODCB15 = 1`

LED1 macro makes changing of LED1 pin assignment easier, also improves code clarity.

`DELAY_MS(ms)` macro is defined in *common\pic24_delay.c* in the book source distribution, `ms` is a `uint32` value.

Data Sheets ...

PIC24 Datasheets

- You MUST be able to read the PIC24 datasheets and find information in them …
  - The notes and book refer to bits and pieces of what you need to know, but DO NOT duplicate everything that is contained in the datasheet
- The datasheet chapters are broken up into functionality …
  - I/O Ports
  - Timer0
  - USART

PIC24 Datasheets

- In each chapters are sections on different capabilities …
  - I/O ports have a section on each PORT
- The PIC24 Family reference manual has difference sections for each major subsystem
- The component datasheet for the PIC24HJ32GP202 has summary information …
  - You will need to refer the family reference manual most often.

Processor Resets ...

Power-on Reset Behavior and Reset Sources

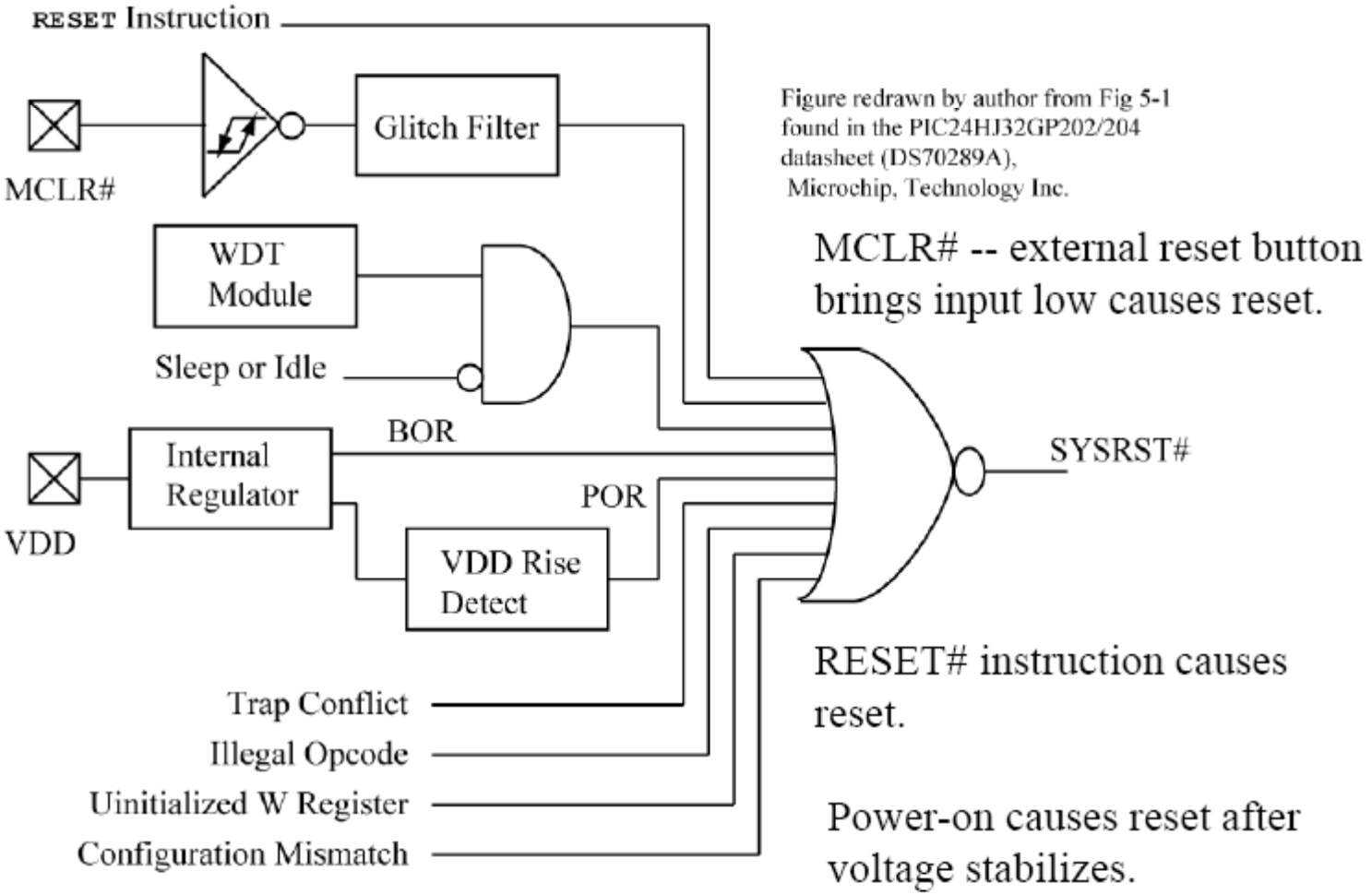


Figure redrawn by author from Fig 5-1 found in the PIC24HJ32GP202/204 datasheet (DS70289A), Microchip, Technology Inc.

MCLR# -- external reset button brings input low causes reset.

RESET# instruction causes reset.

Power-on causes reset after voltage stabilizes.

What Type of Reset Has Occurred?

Flag Bit	Set by:	Cleared by:
TRAPR (RCON<15>)	Trap conflict event	POR, BOR
IOPUWR (RCON<14>)	Illegal opcode or initialized W register access	POR, BOR
CM (RCON<9>)	Configuration Mismatch	POR,BOR
EXTR (RCON<7>)	MCLR# Reset	POR
SWR (RCON<6>)	`reset` instruction	POR, BOR
WDTO (RCON<4>)	WDT time-out	`pwrsav` instruction, `clrwdt` instruction, POR,BOR
SLEEP (RCON<3>)	`pwrsav #0` instruction	POR,BOR
IDLE (RCON<2>)	`pwrsav #1` instruction	POR,BOR
BOR (RCON<1>)	BOR	n/a
POR (RCON<0>)	POR	n/a

Note: All Reset flag bits may be set or cleared by the user software.

Bits in the RCON special function register tell us what type of reset occurred.

Watchdog Timer ...

Watchdog Timer

- We touched upon this in Microprocessors B …
  - With a new processor …
    - We need to investigate it in a little more detail
- Do you recall what is a Watchdog timer is/does?

Watchdog Timer

- ***Error-Recovery:*** If the CPU starts a hardware operation to a peripheral … and … waits for a response …

  – The WDT can break the CPU from an infinite wait loop by resetting the CPU if a response does not come back in a particular time period

- ***Wake From Sleep Mode:*** If the CPU has been put in a low power mode (clock stopped) … then … it can be used to wake the CPU after the WDT timeout period has elapsed

Watchdog Timer Specifics

- Using free-running RC oscillator, frequency of about 32.768 kHz, runs even when normal clock is stopped
- *Watchdog timeout* occurs when counter overflows from max value back to 0.
- The timeout period is …

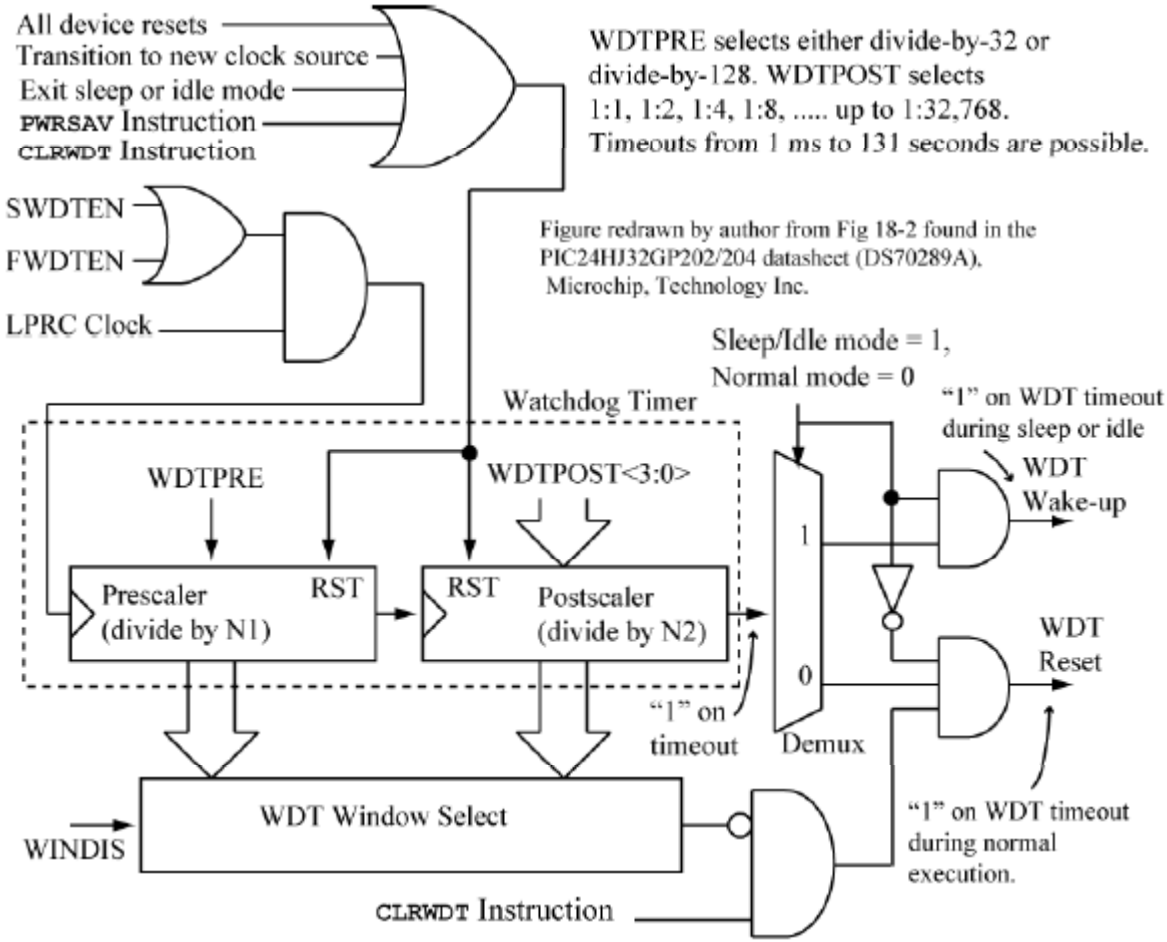
  WDT timeout = 1/32.768kHz x (WDTPRE) x (WDTPOST)

- Times from 1 ms to 131 seconds are possible, bootloader firmware set for about 2 seconds

Watchdog Timer Specifics

- A WDT timeout during normal operation …
  - RESETS the PIC24
- A WDT timeout during sleep or idle mode (clock is stopped) …
  - Wakes up the PIC24 and resumes operations
- The clrwdt instruction clears the timer prevents overflow

Watchdog Timer



Power Saving Modes ...

Power Saving Modes

- **Sleep** … **Idle** … and … **Doze**

- **Sleep:**
  - Main clock stopped to CPU and all peripherals
  - Can be awoke by the WDT
  - Use the pwrsav #0 instruction

- **Idle:**
  - Main clock stopped to CPU but not the peripherals (UART can still receive data)
  - Can be awoke by the WDT
  - Use the pwrsav #1instruction.

Power Saving Modes

- **Doze:**
  - Main clock to CPU is divided by Doze Prescaler (/2, /4, … up to /128)
  - Peripheral clocks unaffected, so CPU runs slower, but peripherals run at full speed – do not have to change baud rate of the UART

Current Measurements

Mode	PIC24HJ32GP202 @40MHz (mA)	PIC24FJ64GA002 @16 MHz (mA)
Normal	42.3	5.6
Sleep	0.030	0.004
Idle	17.6	2.0
Doze/2	32.2	4.0
Doz	17.9	2.0

Doze current(/N mode) = Idle current + (Normal current − Idle current)/N

The idle current is the base current of the chip with the CPU stopped and the clock going to all of the peripherals. So any doze mode current adds to this base.

Programming / Software

MPLAB PIC24 *C Compiler*

- We will be using C for our labs …
  - At times we may need to use assembly language
- Will use the MPLAB PIC24 C Compiler from Microchip
  - Excellent compiler, based on GNU C … which generates very good code

Programming/Software

- We needed to change compilers this semester because …
  - Hi-Tech C we were using does not support the PIC24
- C is basically C … however …
  - The processor has …
    - More features
    - Different nomenclature
- That we will need to get use to

“C” Commands

- We will discuss them as they come up in the material

Lab

Peer Review of Software ...

Peer Review of Software Developed

- Will start the review in a few weeks …
  - Once we get into our labs

Lab Overview …

LAB OVERVIEW

- Simulation using software is not an acceptable alternative to breadboarding

- Benefits of hands-on labs/breadboarding …

  - Use of …
    - Components
    - Test equipment
      - Knowledge of Test equipment is a foundation for hardware troubleshooting
  - ** Learn troubleshooting techniques
  - ** Will greatly enhance the class material
  - Solving Lab Problems will enforce the course material

LAB OVERVIEW

- Basic lab knowledge/techniques
  - Use of a breadboard
  - Learn the identification systems for components
    - Resistors
    - Capacitors
    - Integrated circuits
  - Application of data sheets

LAB OVERVIEW

- **Problems encountered during lab performance …**
  - Knowledge gained from troubleshooting

LAB OVERVIEW

- Lab grade …
  - Lab proficiency 20 points
  - Lab Report Format 20 points
  - Lab Notebook 20 points
  - Technical adequacy 40 points
  - Late deductions as much as 30 points

- Explanation on why a lab could not be completed

- Lab preparation
    - Need to work through the lab prior to class

- Lab Results …
  - Record in your lab notebook events/results
  - Write on what you did during the class …
    - Not what you did after the lab

LAB OVERVIEW

- Things you should bring to the lab …
  - Laptop (highly recommended as the lab computers get heavily used)
  - Flash drive
  - PICkit™ 2
  - A container for your board, parts, tools, etc.
  - Tools
    - Wire strippers
    - Wire cutters
    - Jeweler's screwdrivers
    - Screwdriver
  - A copy of the lab (available on the web)

LAB OVERVIEW

- Lab Reports
  - Report form for each lab will be available on the web page
  - Electronic report submission is an alternative to hard copies …
    - PDF format preferred … talk to me about other formats
    - Send via email NLT than 11:59 PM on the due date
  - Send your programs to me via email

- Lab results will usually be due 2 weeks after completion of the lab (as indicated on the syllabus)

- Labs will be available for downloading from the website NLT Monday evening

Labs ...

Lab

- Lab #1 will start next week
  - I will post it on the Course Web page by next Monday night

Next Class …

Next Class Topics

- General-purpose I/O … LED/switch I/O, State Machine Programming
- Read the material (see homework) …
  - Today's portion of the book
  - Next week's topics
- Start Lab #1

HomeWork

Homework

- Read …
  - Material covered in today's lecture …
    - Chapter 3, pages 53 – 61
    - Chapter 8, pages 241 – 249, 252 – 261, 266 – 287
  - Material for next week's lecture …
    - Chapter 8, pages 287 – 312

Time To ...

Start the Lab ...

Lab

- No lab scheduled for this week …

References

1. None