CS-424/580A Microcontrollers and Robotics

CS-424/580A Microcontrollers & Robotics

- Professor Richard R. Eckert
  - EB-N6, 777-4365
  - Office hours: W 10-11:30 A.M., R 1-2:30 P.M.
  - Email: reckert@binghamton.edu
  - Web Page: www.cs.binghamton.edu/~reckert/
    - Link to CS-424/580A
  - Class listserv
    - CS424-L@listserv.binghamton.edu
    - To be activated soon
- CA: Ryan Zielinski
  - Email: rzielin1@binghamton.edu
  - Office hours: TBA

Course Content

- Microcontrollers
  - Architectures, instruction sets, and programming
    - Microchip Technology's PIC microcontrollers
      - PIC18F452
  - Control of alphanumeric LCD displays
  - Digital and analog I/O ports
  - External memory interface
  - A/D, D/A conversion, sensors
  - Motor control, PWM
  - Timers
  - Interrupts
  - Serial I/O
    - USARTs

- Robotics
  - LEGO Mindstorms RCX
  - Programming the RCX with NQC
    - Motor/sensor control, Sound, Multitasking, Timers, IR communication, Data logging
  - Behavior architectures
  - Robot navigation
  - Robot vision control
  - Programming with Lejos Java
  - Open-Robot
    - Telerobotics
    - Interfacing with Microsoft .NET
    - Modifying firmware with PIC C Compiler and ICD
    - Robot communications
  - Robot competition

Lab

- Most important part of course
- Every Friday, 1:10-4:10 P.M., LNG-210
  - We will meet this Friday
- Students work in teams of three
- One report per team for lab experiments
- 1st half of course: Microcontroller experiments
  - QuikFlash & QuikProto boards
    - PIC 18F452 microcontroller
- 2nd half, robot experiments
  - LEGO Mindstorms RCX
  - Open-Robot

The Microcontroller

- Common component in modern electronic systems
- Computer on a single chip
  - Microprocessor-based device (the "core")
  - Completely self contained with memory and I/O on chip
- Primary role:
  - Provide inexpensive, programmable logic control and interfacing to external devices
    - Monitor external input, receive input from sensors
    - Regulate and turn devices on or off
    - Very commonly used in robots
- Usually <u>embedded</u> in the systems they control

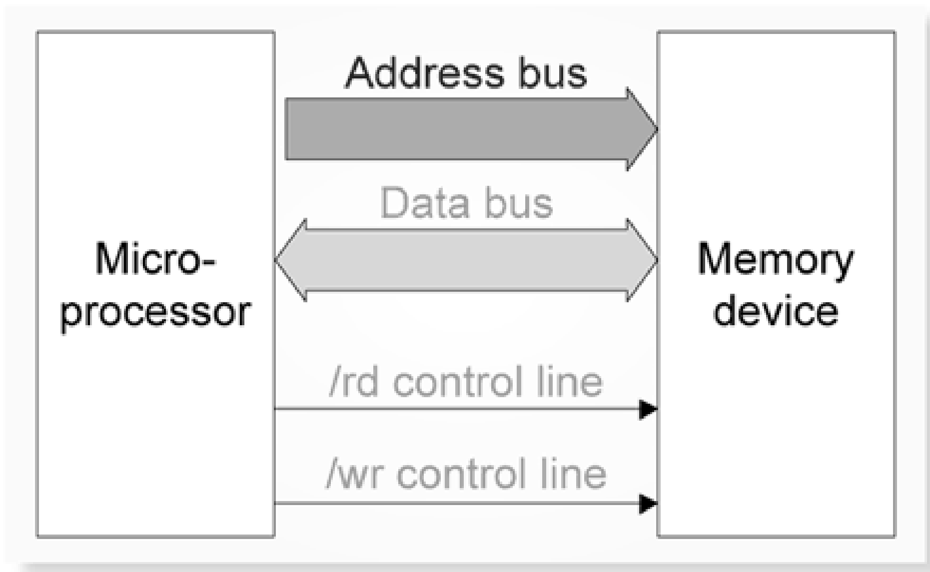
Use of Microcontrollers

- Greatly outnumber conventional CPUs used in PCs
- Used in a wide variety of electronic systems:
  - Automobile systems
  - PC keyboards and printers
  - Electronic measurement instruments
  - Mobile phones
  - TV, radio, CD/DVD players, tape recording equipment
  - Hearing aids
  - Security alarm systems
  - Microwave ovens
  - Remote controllers
  - Food dispensers
  - Many, many more

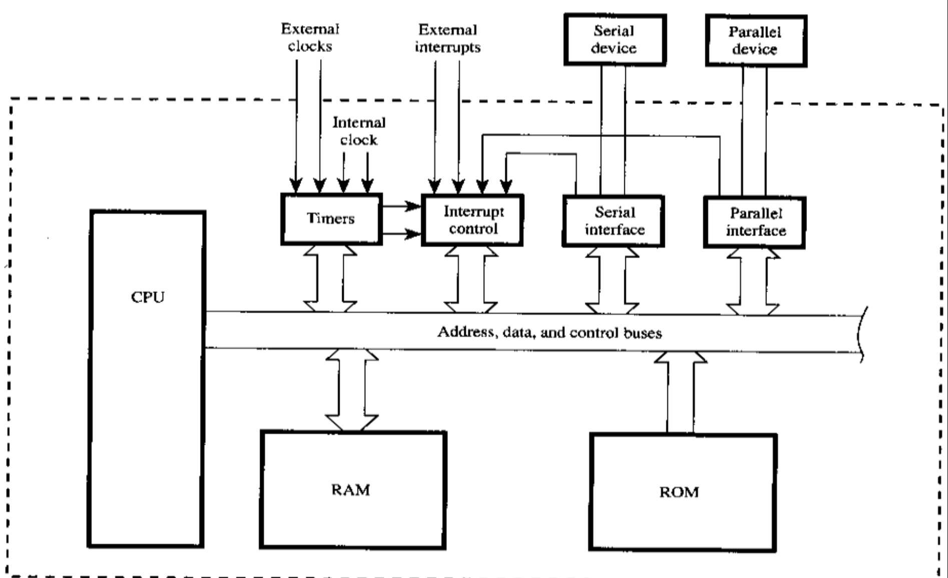
Microprocessor

- Programmable device that integrates many useful functions into a single IC package
- The CPU of a computer on a chip
- Some functions:
  - Execute a stored set of instructions to carry out user-defined tasks
  - Access external memory and I/O chips to read/write data from/to memory and I/O devices

Organization of a Microprocessor-based Microcomputer System



Microprocessor-based Computer System Details



Detailed block diagram of a microcomputer system

Components inside dotted line are found in most microcontrollers

Input/Output

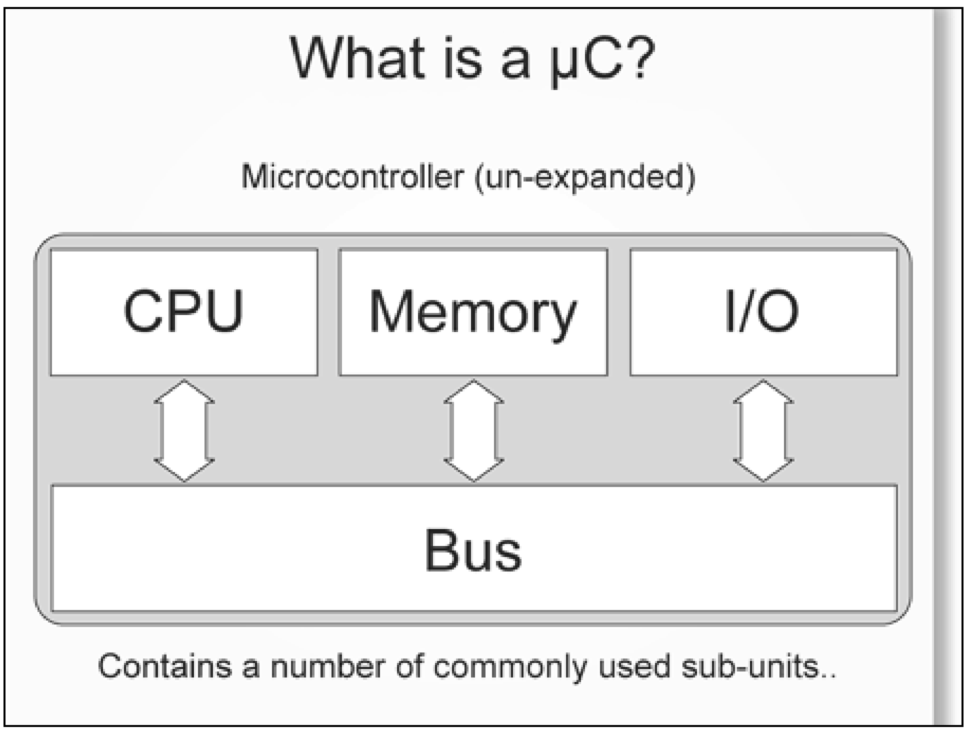
- Digital I/O Ports
- Serial vs. Parallel
- Asynchronous vs. Synchronous
- Programmable Parallel Ports (e.g., Intel 8255 PPI)
  - Direction, handshaking conventions, etc.
- Serial Interface (e.g., Intel 8251 USART)
- Analog I/O
  - A/D, D/A Converters (e.g., 804, 1408)
- Timers (e.g., Intel 8254 PIT)
- Programmed vs. Interrupt-driven I/O
  - Interrupt Controllers (e.g. Intel 8259 PIC)

Memory

- In order of increasing "writeability" and volatility
  - ROM
  - PROM
  - EPROM
  - EEPROM
  - Flash (non-volatile read/write memory)
  - Static RAM
  - Dynamic RAM

Basic Microcontroller

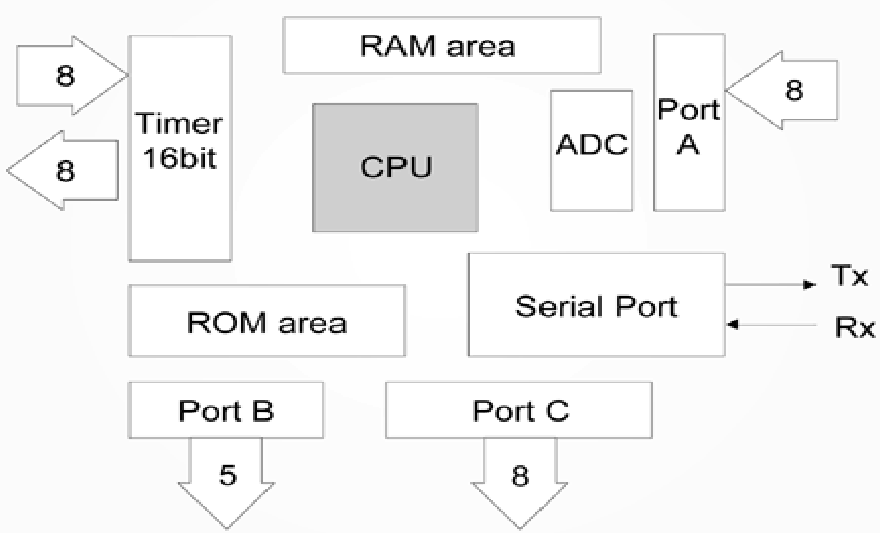
- Integrates most of the components of a microcomputer system onto a single chip
- Just need to add power and clocking
- Following components usually included:
  - CPU Core
  - I/O
  - Memory
    - PROM, EPROM, or Flash for programs and nonvolatile data
    - RAM for volatile data



More Complete Microcontroller

- CPU, I/O, Memory
- Timer module
  - perform tasks for specified time periods, output pulses of determined length, measure time intervals
- Digital I/O
- Serial I/O (USART)
- Analog I/O (D/A, A/D)
  - e.g. to receive data from sensors and control motors and other analog devices
- Interrupt controller
- External memory interfaces to expand on-chip memory
- Built-in monitor/debugger program
- Support for external peripherals
  - e.g., I/O and bus extenders
- Often other support devices often used with them
  - In-circuit debuggers and emulators

A Single Chip Microcontroller



CPU: The processing module of the microcontroller

Differences between Microprocessors and Microcontrollers

- Design:
  - mC: Designed to control I/O devices
  - mP: Designed to process large amounts of data fast in large computer systems
- Instruction Sets:
  - mP: processing intensive ✍
    - Powerful, complex instructions w/ many addressing modes
    - Large instruction size
  - mC: Instructions to control I/O ✍
    - Small instruction sets, set/clear bits, Boolean operations
    - Compact instructions
      - Control program must fit in small on-chip PROM

- Hardware and Instruction Set Support:
  - mC: Built-in I/O operations, event timing, interrupt priority schemes
  - mP: Requires external support (programmable controller chips) for these activities
- Bus Widths:
  - mP: Very wide
    - Large memory address spaces ✍ wide Address Bus
      - e.g., 4 Gigabyte memory ✍ 32 bit wide Address Bus
    - Lots of data transferred very fast ✍ wide Data Bus
      - (32, 64, 128 bits common)
  - mC: Relatively narrow
    - Memory size: kilobytes ✍ 8/16 bit Address Bus
    - Data Bus typically 4, 8, 16 bits wide

- Clock Rates
  - mP: Very fast for fast processing of large amounts of data
    - Typically > 1 GigaHertz
  - mC: Relatively slow to control slow I/O devices
    - Typically 10 KHz -10 MHz
- Cost
  - mP: High
    - Typically > $100.00
  - mC: Low
    - 4 Bit: < $1.00
    - 8 Bit: $1.00 to $10.00
    - 16 Bit: $10.00 to $20.00
    - Even less in bulk quantities

Microcontroller Software

- Programming
  - Usually in core CPU's native assembly language
  - Sometimes HLL support available (C, Basic)
  - Assemblers/Linkers often provided by mfg.
  - C Compilers:
    - Sometimes free – tend to be buggy
    - Better ones are expensive
  - Programming environments
    - Often very unfriendly (not Microchip's MPLAB)
      - Powerful IDE for assembly lang. pgm. development
      - MPLAB SIM simulator for debugging

Microcontroller Pgm. Downloading

- Program Development usually done on a PC
- SW tools must produce a file that can be downloaded to the mC's PROM or Flash memory
  - Several standard formats
    - Intel Hex format most common
- EEPROM burner often needed
  - Expensive quartz glass window
    - UV EPROM eraser required
- If mC has Flash memory, it's much easier
  - Can be reprogrammed with resident monitor program
  - Often on-chip USART to communicate w/ the PC
  - In-circuit debuggers and emulators can be used

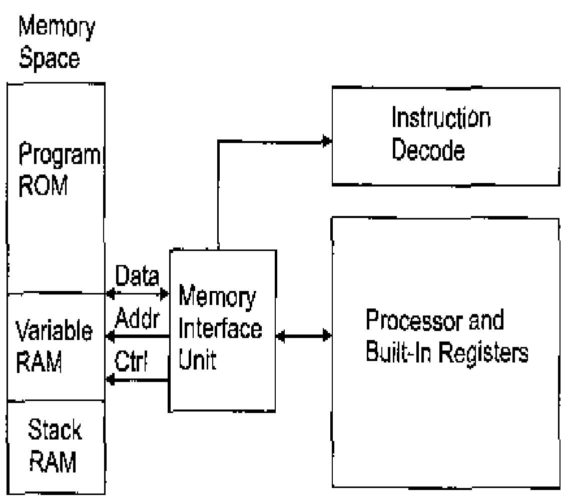
Monitor Program

- On-chip program that communicates with PC software
- Typically uses a serial port to talk to PC terminal emulation program
- Capabilities vary
  - Used to download programs, but often includes a debugger
    - Examine/change registers, memory
    - Single step, set break points
- We'll be using the TeraTerm Pro Terminal program on a PC to communicate with the QuikBug monitor burned into the PIC18F452 mC on our QuikFlash microcontroller boards

Microcontroller Architectures

- Princeton (Von Neumann) Architecture
  - – All memory space on same bus
  - – Instructions and data treated in same way
    - Possible bottleneck between instruction fetches and data fetches
  - – Simple processor design
    - Only one memory interface
    - More reliable since fewer things can fail
    - RAM used for both data ans instructions ✍ greater flexibility in SW/OS design
  - – Intel 80x86 (Pentium) mP, Motorola 68HC11 mC are examples

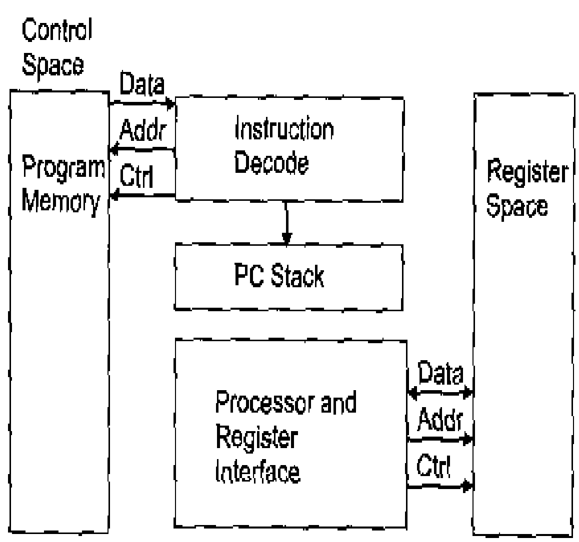
Princeton Architecture



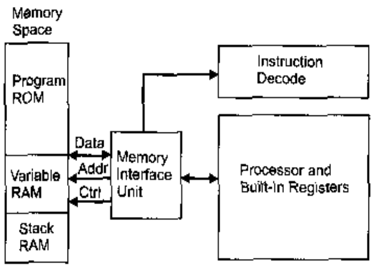
Princeton architecture block diagram

- Harvard Architecture
  - – Code and data memory storage areas are on different busses
  - – Potentially more efficient
  - – Instructions execute in fewer cycles
    - Instructions and data can be fetched simultaneously
    - Greater instruction parallelism
  - – More complex processor design
  - – Example: Microchip's PIC microcontrollers

Harvard Architecture



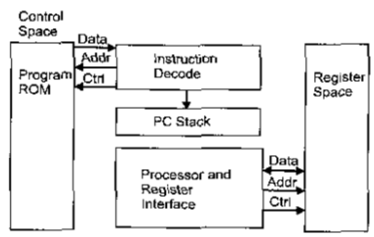
Harvard architecture block diagram



Cycle 1:
- Read Instruction

Cycle 2:
- Read Data out of RAM and Put into Acc.

"move Acc, Reg" in Princeton Arch.



Cycle -1:
- Complete Previous Instruction
- Read the "move Acc, Reg" Instruction

Cycle 1:
- Execute "move Acc, Reg" Instruction
- Read next Instruction

"move Acc, Reg" in Harvard Architect

CISC vs. RISC

- CISC
  - Complex Instruction Set Computer
- RISC
  - Reduced Instruction Set Computer

CISC

- Many instructions in the instruction set
  - Can be very powerful and serve very special purposes
- Many addressing modes
- Can do complex operations with one instruction
  - But many are used very infrequently
- Many are very long (many bytes) and require many clock cycles
- Example: Intel 80X86
- Control Unit (Instruction Decoder) must be very complex, occupying much of chip area
  - Less space for registers
  - More access to memory required
  - Slower

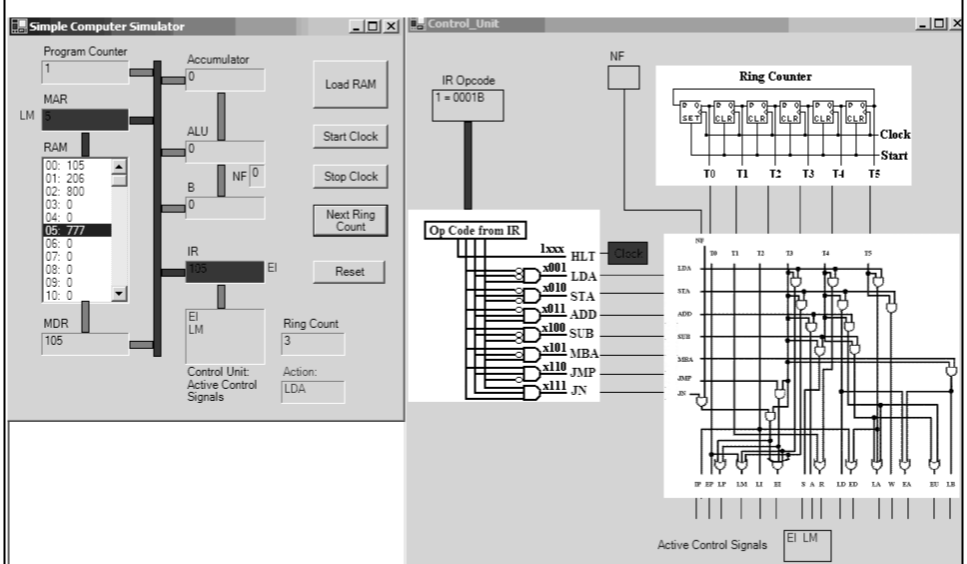
RISC

- Few instructions in the instruction set
  - Simple instructions
  - Short and fast
  - Often instructions are “orthogonal”
    - All access registers in same way
- Few addressing modes
- Example: PIC and many other microcontrollers
- Some mPs and mCs offer both CISC and RISC features
- Simpler (smaller) control unit
  - More space for more registers
  - Less access to memory
  - Faster

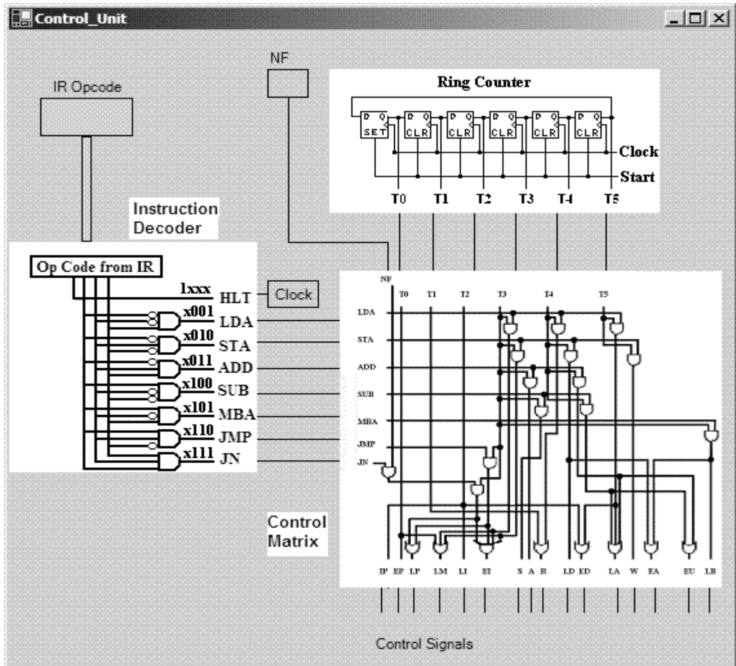
Internal Organization of a Computer's Control Unit

- Hardwired
  - – Control signals required to execute instructions generated by logic gates in a "control matrix"
    - Faster, but less flexible
- Microprogrammed
  - – A processor within the processor causes "control words" to be fetched from a control ROM
    - Bits are control signals
    - Greater flexibility in instruction set design
    - Easier to design (SW vs. HW)
    - But slower

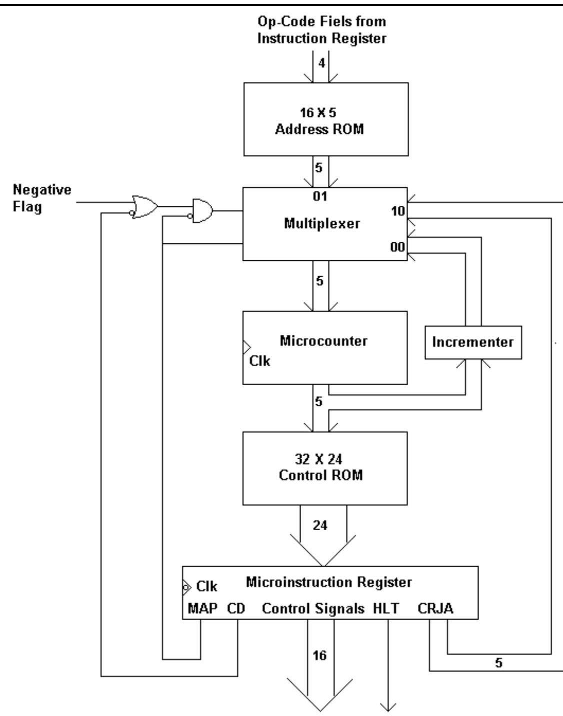
Computer with Hardwired Control Unit



A Hardwired Computer Control Unit



Microprogrammed Computer Control Unit



Some Examples of Microcontrollers

- Intel 8051
- Microchip’s PICmicro
  - We’ll use the PIC18F452 in first half of course
- Motorola 68HC11
- Hitachi H8
  - Microcontroller in the LEGO MindStorms RIS RCX used in the second half of the course

Intel 8051 Family

- Introduced in late 1970s
- At the heart of biggest variety of micros on earth
- CISC with Harvard Architecture
- Some Pros:
  - Powerful bit manipulation instructions
  - Part of RAM & many registers bit-addressable
  - Multiply/divide instructions
  - At least two 16-bit timers
  - UART capable of 500 kb/s at 16 MHz
  - Lots of internal RAM
  - Straightforward code addressing – always starts at 0
  - Bidirectional I/O easy
  - Excellent free assembler and C compiler available

Intel 8051

- Some Cons:
  - Quirky instruction set
  - Confusing variety of internal/external memory spaces
  - I/O design prevents true floating inputs
  - Not very fast due to clock division
  - Access to external data very weak
    - Single pointer register only
  - Watchdog timer and Brownout Reset not standard

Microchip's PICmicros

- Modern PIC18F452 is the mC on the QuikFlash board used in first half of the course
- RISC with Harvard Architecture
- Some Pros:
  - Simple instruction set
  - Robust hardare
    - Lots of devices incorporated on chip
  - Excellent support
    - Microchip's MPLAB free development environment
  - Wide range of sizes and peripheral choices
  - Many language and tool choices

PICmicros

- Some Cons:
  - – 4 times slower at any given clock speed
  - – Relatively weak instruction set
  - – Limited code and data addressing
  - – Complex “banked” addressing
  - – Accumulator-based data processing
    - Single “working” accumulator

Motorola 68HC11

- Some Pros:
  - – Competitive with Intel 8051
  - – Princeton architecture is easy to understand
  - – Powerful, easy-to-use instruction set
  - – Popular parts are available and cheap
  - – Good support for C compilers
  - – Good support from Motorola
- Some Cons:
  - – Weak timers
  - – Old technology
  - – Evaluation board (EVB) expensive and unreliable
  - – Not price-competitive
  - – Slow

Hitachi H8

- Used in LEGO MindStorms RCX
- Similar in architecture to the 68HC11
- Some Pros:
  - Flash memory
  - Powerful instruction set
  - HLL suport:
    - C, Java, Forth, other language compilers free
  - Wide range of 8 and 16 bit devices
  - Nice evaluation/programming boards available
  - Abundant timers
- Some Cons:
  - Fairly expensive