Comparison Report of AVR architecture to PIC architecture

Saige Liu V00812068

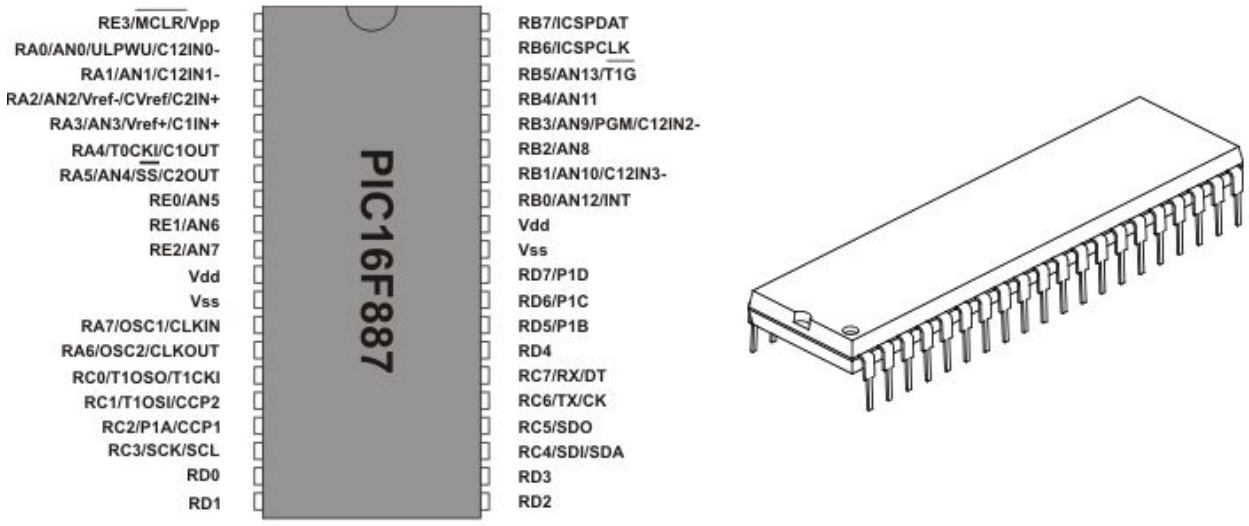
(Comparison is based on specific boards, Arduino Mega 2560 and PIC 16F877)

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560.

PIC (Peripheral Interface Controller) is a family of modified Harvard architecture microcontrollers made by Microchip Technology [1]. And PIC 16F877 is one of the family.
Advantage of PIC
Small instruction set to learn (only 35 instructions).
RISC architecture
Built-in oscillator with selectable speeds
Easy entry level, in-circuit programming plus in-circuit debugging PICkit units available for less than $50
Inexpensive microcontrollers
Wide range of interfaces including I^2C, SPI, USB, USART, A/D, programmable comparators, PWM, LIN, CAN, PSP, and Ethernet
Availability of processors in DIL package make them easy to handle for hobby use.[1]

CISC vs RISC
AVR and PIC are both RISC-based processor architectures. Only a few CISC architecture chips are used in the contemporary Computer Science area. Since the demand of software technology is increasing and the price of RAM memory is decreasing, and the RISC use of RAM and emphasis on software is ideal. Today, the intel x86 is the only one that still retains CISC architecture [2].

Registers of PIC 16F877



Graph 1, Graph of the registers of PIC

The PIC 16F877 has registers from RA0 to RA7, from RB0 to RB7, from RC0 to RC7, from RD0 to RD7, from RE0 to RE7, total 40 registers [3].

They are general purpose registers (Those registers are called pins in PIC), but not only for general purpose. For example, The number two pin has four functions

RA0/AN0/ULPWU/C12IN0	Number 2	RA0	General purpose I/O port A
		AN0	A/D Channel 0 input
		ULPWU	Stand-by mode deactivation input
		C12IN1	Comparator C1 or C2 negative input

Registers of AVR

Learned from classes, AVR has 32 registers, From R0 to R31,and they are all general purpose registers.

R0 to R15 is general purpose registers. But the AVR microcontroller general purpose registers from R0 to R15 could not be used for loading a constant directly

The “ldi” instruction(Set value to the register)can not use registers from R0 to R15.

The registers R26:R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the data space.

The distribution of AVR registers are easier to understand and define.

Program Counter

PIC 16F877

Program memory is 8k x 14. Program address fetch bus is of 13-bit. There are thus 8192 program instruction addresses.

Program counter is thus of 13-bit word pointing to one of the 8 k addresses in program memory.

Total are 8192 addresses in flash in PIC 16F877 as Program counter is of 13 bit.[4]

Arduino Mega 2560

The ATmega2560 Program Counter (PC) is 17 bits wide, thus addressing the 128K program memory locations.[5]

In this case,Arduino Mega 2560 has more Program counter bits than PIC 16F877, but the two series are actually neck and neck. Different models have different bits of program counter, so it depends the models chosen.

Instruction

In AVR, in order to add two constants together.

```
ldi r16, 0x36
ldi r17,0x42
add r16, r17
```

In PIC 16F877, add two constants together:

```
MOVLW 36h
//move value 0x36 into the w
//The W register is a general register in which you can put any value that you wish. Once you have //assigned a value to W, you can add it to another value, or move it. If you assign another value to //W, its contents are overwritten.
MOVWF    FSR
//move the value in w to f register.
MOVLW    42h
ADDWF    FSR, 0
//add value in f and w registers together and put into f
```

Instructions of AVR is simple and easy to understand and write, while the PIC can only do operation on W register, so it's more complex to come up with the correct instructions to complete same job.

Packages

When you're doing one off/prototype projects, you'll want a chip thats easy to work with, that
Means means DIP packaging.
Both AVR & PIC have lots of chips in DIP packages, so it's a tie.

Price

Compare the prices of similar chips in the PIC and AVR series.
8-pin: PIC12F629 ($1.29) v. ATtiny13 ($1.40)
~20-pin: PIC16F628 ($3.35) v. ATtiny2313 ($2.26)
40-pin: PIC18F452 ($10.35) v ATmega32 ($8.17)
Prices of PIC boards are a little bit higher than AVR's.

Full Development Boards

For PIC

PIC Kit 1: this is a relatively new little USB powered board that comes with a bunch of LEDs, a prototyping area and an RS232 converter.

Pros: Simple, straight to the point, useful manual, and $40
Cons: However, no buttons, or anything else other than a few LEDs and only supports a handful of chips.

For AVR

STK500: An old stand-by. Tons of stuff including LEDs, buttons, eeproms, variable oscillators, power supplies etc. $80.
Pros: Full featured, the end-all-be-all of dev boards
Cons: Twice as expensive, serial port not USB so you need a converter for Macs or some PCs
PICKit1 is much cheaper than STK500, so PIC wins this time[7].

Overall, AVR is more idea for beginner to start to do assembly languages, and it's instructions are much more easy to understand.

References

[1] http://en.wikipedia.org/wiki/PIC_microcontroller
[2] http://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/
[3]http://www.mikroe.com/chapters/view/2/chapter-1-pic16f887-microcontroller-device-overview/
[4]http://www.dauniv.ac.in/downloads/MController_PPTs/MicroC2_eCh13L02PICArchitecture.pdf
[5] http://www.wvshare.com/article/AVR-1-1-47.html
[6] http://www.atmel.com/images/doc2549.pdf
[7]http://www.ladyada.net/library/picvsavr.html

Related links of information of PIC

http://ww1.microchip.com/downloads/en/DeviceDoc/sect3_1.pdf
http://ww1.microchip.com/downloads/en/DeviceDoc/31029a.pdf
http://www.ladyada.net/library/picvsavr.html
http://ww1.microchip.com/downloads/en/DeviceDoc/sect2.pdf
http://www.mikroe.com/chapters/view/10/chapter-9-instruction-set/
http://www.microcontrollerboard.com/pic_memory_organization.html
http://www.microcontrollerboard.com/pic_microcontroller.html#PICchoose
http://www.circuitstoday.com/pic-16f877-architecture-and-memory-organization
http://www.engr.du.edu/richard/Classes/Embedded/Refs/S3CEV40_UserGuide.pdf
http://www.hobbyprojects.com/pic_tutorials/tutorial2.html