Introduction

microcontroller

- microprocessor
  - Control unit
  - data path
    - registers
    - arithmetic and logic unit
  - memory
  - I/O

Term

- microprocessor:
  - the implementation often central processor unit runction, or a computer in a 'ingle. large scale integrated (LSI) circuit
- microcomputer:
  - a computer built using a microprocessor and a few other components rorthe memory and 1/0
- microcontroller: -a microcomputer with its memoT)! and va integrated into a single chip.
  - CPU
  - ROM: program and constant data (usually)
  - RAM: variable data
  - I/O Interface
    - timer
    - PWM

computer instruction

- operation
  - mnemonic
- operands
  - may have a source operand and a destination operand

computer architecture

- Neumann architecture
  - CPU: central processor unit
  - memory (ROM and RAM)
  - I/O interface
  - Buses (data, address, control)
- Harvard architecture (DSP)
  - data memory
  - program memory

I/O synchronizatino

- wait ~ ready

AVR Programming

stack

- last in first out
- usually grows from higher address to lower address
- default value of SP is 0x21FF

function

- easy to design(top-down design)
- modularity(readability and maintainability)
- for reuse
- save code size and memory space

function call

- rule 1
  - using stack(if need to be saved in several places) or register for parameter passing
- rule 2
  - parameter can be passed by value or reference(efficient, may be changed in subroutine)
- rule 3
  - callee saves conflict register
- rule 4
  - local variables and parameter need to be stored contiguously on the stack

calling conventions

- C calling convention
  - caller cleans up the stack
- Pascal convention
  - callee cleans up the stack
- fast calling convention
  - using register to pass parameters
  - callee cleans up the stack

stack frame

- return address
- conflict register
- local variable
- parameter

- empty
- reserve space for local variable

```
; prologue
...
in YL, SPL
in YH, SPH
sbiw Y, 8
out SPH, YH
out SPL, YL

std Y+1, r22
std Y+2, r23
std Y+3, r20
std Y+4, r21

; function body
...
ldd r21, Y+4
ldd r20, Y+3
ldd r17, Y+2
ldd r16, Y+1
...

; epilogue
adiw Y,8
out SPH, yH
out SPL, YL
...
ret
```

assembly program structure

- assembler directive

```
.def symbol=register
.equ symbol=expression
.set symbol=expression  ; re-definable

.dseg ; data memory
lable: .byte expr ; a number of byte for variable
ld
lds
st
sts
.cseg ; code memory
in code memory
lable: .db    ; one byte
lable: .dw    ; two byte(little endian)
lpm   ; address in code memory is word address, however, lpm is byte address

ldi ZH, high(table_1 <<1)
ldi ZL, low(table_1 <<2)
lmp r16, Z
```

- assembler expression

```
low()
high()
```

- macro

assembly

- pass one
  - syntax errors
  - expand macro
  - record all label in a symbol table
- pass two
  - substitute for symbols
  - assemble each instruction
- absolute assembly
- relocatable assembly
  - generate object file, with some address may not be resolved
  - a linker program is needed to resolve all unresolved address

Memory access

Assembly process

memory

- Program memory(flash, 16bits)
  - Interrupt Vectors
  - program
- Data memory(SRAM, 8bits)

Name	Address	Comments
32 Registers	0-1F	
64 I/O	20~5F	IN/OUT/SBI/CBI/SBIC/SBIS
	60~1FF	ST/STS/STD/LD/LDS/LDD

416 Ex I/O Reg		
8192 Internal SRAM	0200-21FF	stack
External SRAM	2200-FFFF	optional

- EEPROM

Parallel I/O

- I/O addressing (two byte address)
  - memory-mapped I/O
    - advantage
      - simple design
      - no special instruction
      - scalable
    - disadvantage
      - reduce the memory space
      - decode the full address bus
  - separate I/O (one byte address)
    - less expensive address decoder
    - special I/O instruction are required

Interrupt

External interrupts

Internal interrupts (Timer, counters)

- timer
  - counting time periods
- counter
  - counting the events or sth of this nature
- CPU interaction with I/O
  - Polling
    - software queries I/O devices
    - no extra hardware needed
    - not efficient
  - Interrupt
    - I/O devices generate signals to request services from CPU
    - need hardware
    - efficient
- Interrupt Services
  - Global interrupt enable is cleared and all interrupt are disabled
  - I-bit is set when returned
  - it will always execute one more instruction before any pending interrupt is served
- ISR structure
  - Prologue
    - push stack
  - Body
  - epilogue
    - pop stack
    - return
- REset in AVR
  - power-on reset
  - external reset
  - watchdog reset
  - brown-out reset
  - JTAG AVR reset

I/O devices

switch

- switch bounce(5 to 10 ms):
  - NAND latch denouncer
  - software denouncer
    - wait and see(wait 20 to 100ms and test if it is still low)
    - counter-based approach
      - initialize counter to 10
      - check every ms, if it is low ,decrease otherwise increase the counter
      - poll the switch if the counter is 0 or 20

keypad

LCD

Analog I/O

Analog output

- PWM
  - digitally encoding analog signal level
  - duty cycle(pulse width / period)
- DAC
  - signal conditioning block
    - filter

        - isolation
        - buffer and amplification
- resolution
    - number of bits in the digital value
- DAC structure
    - binary-weighted D/A converter
    - R-2R ladder D/A converter
- DAC specification
    - resolution
    - linearity
    - setting time
    - glitches
        - change from 1-0 faster than 0-1
        - sample and hold
- A/D conversion
    - transducer
    - conditioner
        - isolation and buffering
        - amplification
        - bandwidth limiting
    - ADC
    - processor
- Shannon's sampling theorem
    - when a signal is to be sampled, the minimum sampling frequency must be twice the signal frequency. Nyquist rate.
- ADC structure
    - successive approximation converter
        - from MSB to LSB
        - if D/Ais lower than input, the bit remains set
    - parallel A/D convertor
        - 2^N - 1 comparators
    - two-stage parallel A/D convertor
- ADC specifications
    - conversion time
    - resolution
    - accuracy
        - the smallest signal can me measured
    - linearity
        - best 1/2 LSB
    - aperture time

Analog input

Serial Communication

serial I/O VS parallel I/O

- Parallel I/O need one wire for each bit, the cable can be expensive
- susceptible to reflection and noise

structure

- source -> transmit data buffer -> parallel in/ serial out shift register -> serial in/parallel out shift register -> received data buffer -> destination

synchronous VS asynchronous

- synchronous
    - faster
    - need extra hardware

UAET data formates

- start
- data
- parity
- stop

communication system type

- simplex system
    - data are sent in one direction only
    - simple: sender are slower than receiver, no handshaking are required
- full-duplex system
    - data are transmitted in two directions
- half-duplex system
    - data are transmitted in two directions with only one pair of signal lines

Error detection

- frame error
    - first stop bit
- parity error
- data overrun error
    - if any data is yet read but is overwritten by incoming data frame