Introduction to Electric Vehicles

HW#3
Final project

Joonki Hong
Dept. of Electrical Engineering
Korea Advanced Institute of Science and Technology
joonki@cad4x.kaist.ac.kr

KAIST

Timers

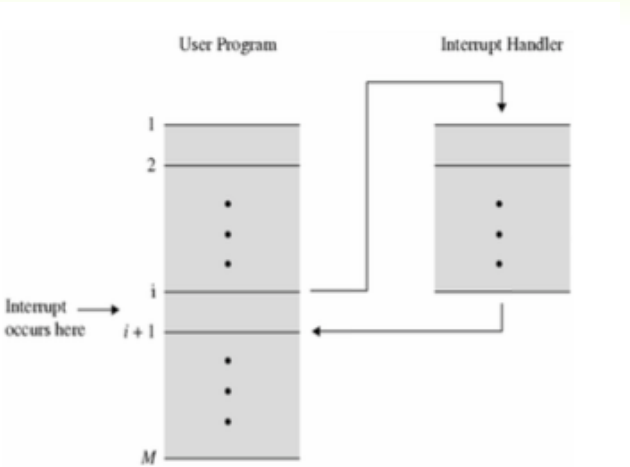
- Timing events
  - Need various frequency timing events: Heart beat
  - Typical time base: e.g. task scheduling: 10ms to 100ms
- The basic unit – counter (up or down)
  - Generates timing events (interrupts or timer output)
    - If overflow or reach 0
    - If match with a preset value
  - Measure time – read counter values (captured)
  - Free running, reset or reload, and compare

Interrupt

- Signal to the processor emitted by hardware or software indicating an event that needs immediate attention
- Hardware interrupt
  - Asynchronous
  - Interrupt request (IRQ)
  - Pressing the keyboard or moving the mouse triggers hardware interrupt
- Software interrupt
  - Exceptional condition in the processor
  - Special instruction in the instruction set

Interrupt

- Handling interrupt
  - Stop the program and prom and save the current state
  - Execute interrupt service routine
    - #include”avr/interrupt.h”
  - Return to program
- Interrupt vector table
  - Interrupt vector is defined to distinguish the source of interrupt
  - Find address of ISR by interrupt vector
- Interrupt priority
  - Decide to select the priority to handle first



Polling vs. interrupt

polling.c

```
main () {
  set_up_pit_polling();

  while (1) {
    while (zds!=1) {
      /* do nothing until timeout */
      }
    clear_zds();
    perform_operation();
    }
}
```

interrupt.c

```
isr() {
  clear_zds();
  perform_operation();
}

main () {
  set_up_pit_interrupt(isr);

  while (1) {
    /* do something useful, isr()
       takes care of the timeout */
  }
}
```

Homework #3

- Implement the myPrintf function
- EEPROM memory access
- Read the data from the data address
- Write the value to the data address
- Memory modification
  - Read one byte from typed memory address
  - Read N bytes from typed memory address
  - Write one byte from typed memory address and one byte data
  - Write N bytes from typed memory address and N bytes data
- Super loop approach
  - Handle user input
  - Support menu functions

Memories

- Read the data sheet pp. 18 - 36

Memory		Mnemonic	AT90CAN32	AT90CAN64	AT90CAN128
Flash	*Size*	Flash size	32 K bytes	64 K bytes	128 K bytes
	Start Address	-	0x00000		
	End Address	Flash end	0x07FFF(1) 0x3FFF(2)	0x0FFFF(1) 0x7FFF(2)	0x1FFFF(1) 0xFFFF(2)
32 Registers	*Size*	-	32 bytes		
	Start Address	-	0x0000		
	End Address	-	0x001F		
I/O Registers	*Size*	-	64 bytes		
	Start Address	-	0x0020		
	End Address	-	0x005F		
Ext I/O Registers	*Size*	-	160 bytes		
	Start Address	-	0x0060		
	End Address	-	0x00FF		
Internal SRAM	*Size*	ISRAM size	2 K bytes	4 K bytes	4 K bytes
	Start Address	ISRAM start	0x0100		
	End Address	ISRAM end	0x08FF	0x10FF	0x10FF
External Memory	*Size*	XMem size	0-64 K bytes		
	Start Address	XMem start	0x0900	0x1100	0x1100
	End Address	XMem end	0xFFFF		
EEPROM	*Size*	E2 size	1 K bytes	2 K bytes	4 K bytes
	Start Address	-	0x0000		
	End Address	E2 end	0x03FF	0x07FF	0x0FFF

Memories

- Flash program memory
  - Write/erase the firmware to flash memory
  - Flash memory space is divided into two sections
    - Boot program section
    - Application program section
- SRAM data memory
  - Volatile memory
  - Fast operation
- EEPROM data memory
  - Non-volatile
  - Slow write/read operation

Variable arguments

- #include<stdarg.h>
- #include<stdlib.h>
- va_list
  - Type for iterating arguments
- va_start
  - Start iterating arguments with a va_list
- va_arg
  - Retrieve an argument
- va_end
  - Free a va_list

EEPROM

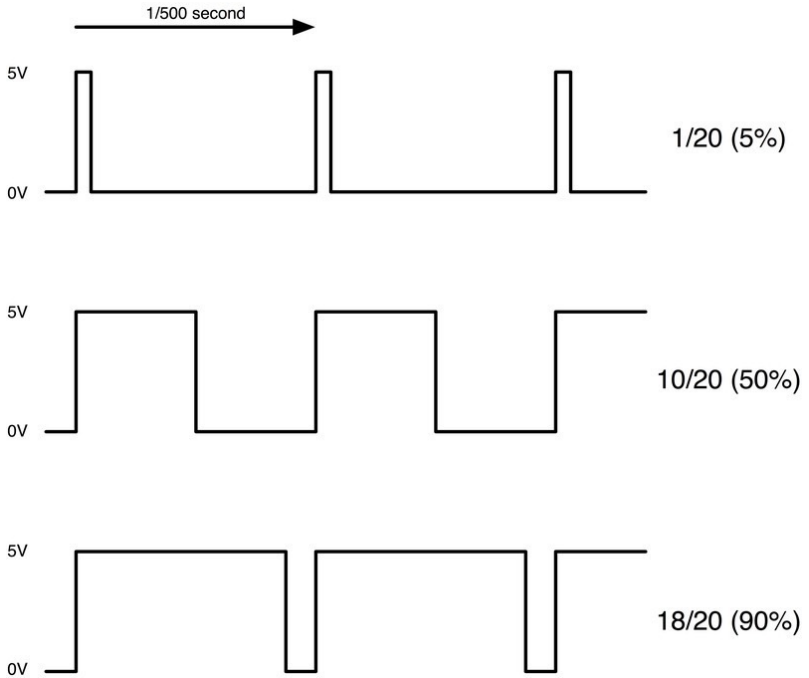
- #include “avr/eeprom.h”
- eeprom_write_byte
  - Write the data to assigned address
- eeprom_read_byte
  - Read the data from assigned address

Final project

- Piezo buzzer
- Frequency control by PWM signal
- Implement a song which you choose

PWM

- Pulse-width modulation (PWM) is a technique used to encode a message into a pulsing signal
- Frequency
  - Number of cycle in one second
- Duty cycle
  - Proportion of 'on' time to the regular interval or 'period' of time



Alternate Port Functions

- AVR microcontroller AT90CAN128 p.71
- Find PWM output port

Port Pin	Alternate Function
PE7	INT7/ICP3 (External Interrupt 7 Input or Timer/Counter3 Input Capture Trigger)
PE6	INT6/ T3 (External Interrupt 6 Input or Timer/Counter3 Clock Input)
PE5	INT5/OC3C (External Interrupt 5 Input or Output Compare and PWM Output C for Timer/Counter3)
PE4	INT4/OC3B (External Interrupt4 Input or Output Compare and PWM Output B for Timer/Counter3)
PE3	AIN1/OC3A (Analog Comparator Negative Input or Output Compare and PWM Output A for Timer/Counter3)
PE2	AIN0/XCK0 (Analog Comparator Positive Input or USART0 external clock input/output)
PE1	PDO/TXD0 (Programming Data Output or UART0 Transmit Pin)
PE0	PDI/RXD0 (Programming Data Input or UART0 Receive Pin)

Piezo buzzer

- Control frequency of the buffer by PWM output