

## Review for Final Exam

- Numbers
  - Decimal to Hex (signed and unsigned)
  - Hex to Decimal (signed and unsigned)
  - Binary to Hex
  - Hex to Binary
  - Addition and subtraction of fixed-length hex numbers
  - Overflow, Carry, Zero, Negative bits of CCR
- Programming Model
  - Internal registers – A, B, (D = AB), X, Y, SP, PC, CCR
- Addressing Modes and Effective Addresses
  - INH, IMM, DIR, EXT, REL, IDX (Not Indexed Indirect)
  - How to determine effective address
- Instructions
  - What they do (MC9S12 Core Users Guide)
  - What machine code is generated
  - How many cycles to execute
  - Effect on CCR
  - Branch instructions – which to use with signed and which with unsigned
- Machine Code
  - Reverse Assembly
- Stack and Stack Pointer
  - What happens to stack and SP for instructions (e.g., PSHX, JSR)
  - What happens to stack and SP for interrupt
  - What happens to stack and SP when program leaves an interrupt service routine

- Assembly Language
  - Be able to read and write simple assembly language program
  - Know basic psuedo-ops – e.g., equ, dc.b, ds.w
  - Flow charts
- C Programming
  - Setting and clearing bits in registers
    - \* `PORTA = PORTA | 0x02;`
    - \* `PORTA = PORTA & ~0x0C;`
  - Using pointers to access specific memory location or port.
    - \* `* (unsigned char *) 0x0400 = 0xaa;`
    - \* `#define PORTX (* (unsigned char *) 0x400)`  
`PORTX = 0xaa;`
- Interrupts
  - Interrupt Vectors (and reset vector)
    - \* How to set interrupt vector in assembly
    - \* How to set interrupt vector in C
  - How do you enable interrupts (specific mask and general mask)
  - What happens to stack when you receive an enabled interrupt
  - What happens when you leave ISR with RTI instruction?
  - What setup do you need to do before enabling interrupts?
  - What do you need to do in interrupt service routine (clear source of interrupt, exit with RTI instruction)?
  - How long (approximately) does it take to service an interrupt?

- Timer/Counter Subsystem
  - Enable Timer
  - Timer Prescaler
    - \* How to set
    - \* How it affects frequency of timer clock
  - Timer Overflow Interrupt
  - Input Capture
  - Output Compare
  - How to enable interrupts in the timer subsystem
  - How to clear flags in the timer subsystem
  - Be able to look at timer registers and determine how timer is set up
    - \* Which channels are being used
    - \* Which are being used for Input Capture, which for Output Compare
  - How to time differences from Timer counts
- Real Time Interrupt
  - How to enable
  - How to change rate
  - How to enable interrupt
  - How to clear flag
- Pulse Width Modulation
  - How to get into 8-bit, left-aligned high-polarity mode
  - Calculate how many clock periods it takes to get desired PWM period (frequency)
  - How to set PWM period (frequency)
    - \* Using Clock Mode 0
    - \* Using Clock Mode 1
  - How to set PWM duty cycle
  - How to enable PWM channel
  - Be able to look at PWM registers and determine PWM frequency and duty cycle

### Serial Communications and the IIC Bus

- Pins used – SDA and SCL
- Difference of use in Master and Slave mode
- IIC serial format for writing to slave
  - \* Start condition, 7-bit slave address,  $R/\overline{W}$ , wait for acknowledge
  - \* Send eight data bits, wait for ACK, repeat, send stop condition
- IIC serial format for reading from slave
  - \* Start condition, 7-bit slave address,  $R/\overline{W}$ , wait for acknowledge
  - \* Receive eight data bits, send ACK, repeat, after receiving last byte, send NACK instead of ACK, send stop condition
- IIC IBAD (Bus Address) register
  - \* Set address when used as slave
  - \* To use as master, write something like 0x01 (any address not assigned to a slave)
- IIC IBFD (Bus Frequency Divide) Register
  - \* Set clock speed to match slave
- IIC IBCR (Bus Control Register) Register
  - \* IBEN — Enable IIC bus
  - \* IBIE — Enable interrupts
  - \*  $MS/\overline{SL}$  Switch to master mode
  - \*  $TX/\overline{RX}$  Switch between transmit and receive
  - \* TKAK — Send an acknowledge
  - \* RSTA — Send an restart (didn't discuss)
  - \* IBSWAI — Specify if IIC clock should operate in WAIT mode (didn't discuss)
- IIC IBSR (Bus Status Register) Register
  - \* TCF — Transmit Complete Flag
  - \* IAAS — Did not discuss
  - \* IBB — Did not discuss
  - \* IBAL — Did not discuss
  - \* SRW — Did not discuss
  - \* IBIF — Interrupt flag. Clear by writing a 1 to this bit.
  - \* RXAK — Did not discuss
- IIC IBDR (Bus Data Register) Register
  - \* Write data to this register to send to slave
  - \* Read data from this register to receive from slave

- A/D Converter
  - How to power-up A/D converter (ATDCTL2)
  - Write 0x05 to ATDCTL4 to set at fastest conversion speed and 8-bit conversions
  - Write 0x85 to ATDCTL4 to set at fastest conversion speed and 10-bit conversions
  - Left justified or right justified, signed or unsigned
  - How to set modes of A/D converter (ATDCTL5)
    - \* Continuous Scan vs. Single Scan
    - \* Multichannel vs. Single Channel conversions
      - Channel for single channel scan
      - Starting channel for multichannel scan
  - How to tell when conversion is complete - ATDSTAT register
  - How to read results of A/D conversions
    - \* Be able to convert from digital number to voltage, and from voltage to digital number (need to know  $V_{RH}$  and  $V_{RL}$ ).

- The MC9S12 in Expanded Mode
  - Getting into expanded mode — MODA, MODB, MODC pins or MODE Register
  - PEAR Register — enable ECLK,  $\overline{\text{LSTRB}}$ ,  $\text{R}/\overline{\text{W}}$  on external pins
  - Ports A and B in expanded mode
    - \* Port A – AD 15-8 (Port A is for data for high byte, even addresses)
    - \* Port B – AD 7-0 (Port B is for data for low byte, odd addresses)
  - E clock
    - \* Address on AD 15-0 when E low, Data on AD 15-0 when E high
    - \* Need to latch address on rising edge of E clock
    - \* On write (output), external device latches data on signal initiated by falling edge of E
    - \* On read (input), MC9S12 latches data on falling edge of E
    - \* E-clock stretch - MISC register
    - \* Disable ROM from 0x4000 to 0x7FFF - MISC register
  - $\text{R}/\overline{\text{W}}$  Line
  - $\overline{\text{LSTRB}}$  line
  - Single-byte and two-byte accesses
    - \* 16-bit access of even address – A0 low,  $\overline{\text{LSTRB}}$  low – accesses even and odd bytes
    - \* 8-bit access of even address – A0 low,  $\overline{\text{LSTRB}}$  high – accesses even byte only
    - \* 8-bit access of odd address – A0 high,  $\overline{\text{LSTRB}}$  low – accesses odd byte only
  - Timing – Be sure to meet setup and hold times of device receiving data
    - \* For a write, meet setup and hold of external device
    - \* For a read, meet setup and hold of MC9S12
  - Timing — Be sure to meet address access time (length of time address needs to be on bus before external device is ready)