EEL 4744C: Microprocessor Applications

Lecture 9

Part 2

M68HC12 Serial I/O
Reading Assignment

• Software and Hardware Engineering (new version): Chapter 15

• SHE (old version): Chapter 11

• HC12 Data Sheet: Chapter 14
Introduction

• Asynchronous serial communication interface (SCI), a.k.a. on-chip UART; 1 on B32, 2 on A4

• Synchronous serial peripheral interface (SPI) for high-speed synchronous serial communication

• Also a byte data link communication (BDLC) module on B32 for SAE J1850 communication in auto. applications (will not be discussed)
Basic SCI & SPI Layout

(a) Asynchronous Serial Communications Interface

(b) Synchronous Serial Peripheral Interface
Asynchronous Serial Communication Interface

- Full-duplex, h/w parity generation, option for single-wire operation

- On-chip generator for standard bit-rates

- Transmitter and receiver double-buffered, operate independently, use same rate and format

- Supports 8- or 9-bit data, variety of flags and interrupts
Asynchronous Serial Communication Interface

• Programming and using SCI includes three components:
  – Initialization of data rate, word length, parity, and interrupting capabilities
  – Writing to SCI data register (being careful not to exceed transmission rate)
  – Reading from SCI data register (being careful to read incoming data before more arrives)
SCI Data

• Two data regs., SCnDRH and SCnDRL (n=0 on B32 and =0,1 on A4)

• SCnDRL reg. is two separate regs. at same address, one for read and one for write

• SCnDRH for MSB of 9-bit data, should be written before SCnDRL for correct data transfer

• If 8-bit data, only SCnDRL used
SCI Data Register

- **R7**: SERIAL RECEIVE DATA
- **R0**: RECEIVER SHIFT REGISTER
- **RxD**: Receive Data
- **T7**: TRANSMITTER SHIFT REGISTER
- **TxD**: Transmit Data
- **T0**: SERIAL TRANSMIT DATA
- **‘SCnDRL’ (READ)**
- **‘SCnDRL’ (WRITE)**
SCI Initialization

• Control bits TE and RE in SCnCR2 to enable transmitter and receiver on each SCI channel

• SCI operation mode must be initialized using SCnCR1
  – In addition to normal SCI operation (default), several other modes available:
    • Wired-OR mode (output pin is open-drain; needs external pullup; used in single-wire system with multiple devices connected together); controlled by WOMS control bit
SCI Initialization

- **Loop mode** (for testing, if rec. source bit RSRC = 0 then rec. connected internally to xmitter and thus xmitter can be disconnected from TxD pin; o/w external)
---

**SCI Initialization**

- **Single-wire mode** (only TxD pin(s) used, and RxD pins available for GP I/O)

---

**Diagram**

- 'SCnCR1'
  - LOOPS=1
  - WOMS=1

- 'DDRS'
  - Bit-1(3)=1

- 'SCDR' (WRITE)
  - 7
  - TRANSMITTER

- 'SCDR' (READ)
  - RECEPTOR

- RSRC=1
  - 'SCnCR1'

- TxD
  - PS1 (PS3)

- V_{DD}

- R

- I/O

---

Dr. Tao Li
• M control bit: (‘0’ means 1 start, 8 data, and 1 stop bit; ‘1’ means 1, 9, and 1)

• H/w to detect idle line (receive line in mark (1) state for more than one character time)

• H/w to generate parity bit; enabled via PE bit, type (even/odd) selected by PT bit (in SCnCR1 register)

• SBR12:SBR0 contain BR bits to select baud rate; supports standard rates (110…38400) and others; set acts as 13-bit divider of bus clock such that baud rate = bus clock / (16 x BR) (see table 11-2)
SCI Status Flags

• Several flags for polling and interrupts; all reset by reading status reg. SCnSR1 (where each flag resides) then reading/writing next byte from/to SCnDRL

• Trans*mit data reg. empty flag (TDRE); set when last char. written to SCnDRL is xfered to output shift reg

• Transmit complete flag (TC); set when last char. completely sent from output shift reg

• Receive data reg. full flag (RDRF); set when data reg. has new data

• Idle line detected flag (IDLE); set when receive line idle
SCI Status Flags

- **Receiver overrun error flag (OR);** set when new char. rec’d before old data read by pgm

- **Noise flag (NF);** h/w takes 3 samples of rec’d signal near middle of each data and stop bit, and 7 during the start bit; if disagreement within any then flag is set

- **Framing error (FE);** set if receiver detects a space (0) instead of mark (1) during stop-bit time

- **Parity error flag (PF);** set if parity incorrect
SCI Interrupts

- All reset for reuse as above

- Transmit interrupt enable (TIE) for interrupts on TDRE flag

- Transmit complete interrupt enable (TCIE) for TC flag

- Receive interrupt enable (RIE) for RDRF or OR flags (must poll in ISR to determine)

- Idle line interrupt enable (ILIE) for IDLE flag
Other SCI Issues

• Sleep and wakeup mode for multidrop applications
Other SCI Issues

• Sleep and wakeup mode for multidrop applications

  – S/w on each receiver puts self to sleep (by setting RWU control bit) until wakeup

  – With each b-cast, receivers all awaken and s/w in them decodes target of message

  – Only addressed station stays awake to receive message, others resume sleep

  – When asleep, all receiver interrupts disabled until awakened by either:
    • WAKE bit = 0, a full char. of idle line (i.e. mark or ‘1’) wakes up receiver
    • WAKE bit = 1, any byte with a one in the MSB wakes it up
Other SCI Issues

• Sleep and wakeup mode for multidrop applications

  – Note: when asleep, only SCI module is, as the CPU can continue to operate

  – CPU can awaken SCI by clearing the RWU bit, although auto. h/w typically used
Other SCI Issues

• SCI can send break char. (10-11 zeros) by pgm. writing ‘1’ into SBK bit; these chars. used on some systems to wake up the receiving end

• For any serial function not enabled, bits of Port S may be used for GP I/O
SCI Example: Initialization

; SCI Register Equates
TE: EQU %00001000 ; Transmitter Enable
RE: EQU %00000100 ; Receiver Enable
TDRE: EQU %10000000 ; TX Data Reg Empty
RDRF: EQU %00100000 ; Rx Data Reg Full
MODE: EQU %00010000 ; Mode bit
PE: EQU %00000010 ; Parity Enable
ODD_P: EQU %00000001 ; Set odd parity
B9600: EQU !52 ; Baud rate = 9600
SC0BDH: EQU $C0 ; Baud rate register
SC0CR1: EQU $C2 ; Control register 1
SC0CR2: EQU $C3 ; Control register 2
SC0SR1: EQU $C4 ; Status register
SC0DRL: EQU $C7 ; Data register

; Subroutine init_sci
; Initialize SCI to 1 start, 8 data and 1 stop bit, odd parity and 9600 Baud.
; Inputs: None
; Outputs: None
; Reg Mod: CCR
init_sci:
    pshd ; Save D reg
    bclr SC0CR1,MODE
    bset SC0CR1,PE|ODD_P
    bset SC0CR2,TE|RE
    ldd #B9600
    std SC0BDH ; Set Baud rate
    puld ; Restore x
    rts
SCI Example: Send Data

; SCI Register Equates
TE:    EQU %00001000 ; Transmitter Enable
RE:    EQU %00000100 ; Receiver Enable
TDRE:  EQU %10000000 ; TX Data Reg Empty
RDRF:  EQU %00100000 ; Rx Data Reg Full
MODE:  EQU %00010000 ; Mode bit
PE:    EQU %00000010 ; Parity Enable
ODD_P: EQU %00000001 ; Set odd parity
B9600: EQU !52 ; Baud rate = 9600
SC0BDH: EQU $C0 ; Baud rate register
SC0CR1: EQU $C2 ; Control register 1
SC0CR2: EQU $C3 ; Control register 2
SC0SR1: EQU $C4 ; Status register
SC0DRL: EQU $C7 ; Data register

; Subroutine sci_out
; Send SCI data
; Inputs:   A register = data to send
; Outputs:  None
; Reg Mod:  CCR
sci_out:

spin:    brclr  SC0SR1,TDRE,spin

; Output the data and reset TDRE
    staa  SC0DRL
    rts
SCI Example: Check RDRF Flag

; SCI Register Equates
TE:     EQU %00001000 ; Transmitter Enable
RE:     EQU %00000100 ; Receiver Enable
TDRE:   EQU %10000000 ; TX Data Reg Empty
RDRF:   EQU %00100000 ; Rx Data Reg Full
MODE:   EQU %00010000 ; Mode bit
PE:     EQU %00000010 ; Parity Enable
ODD_P:  EQU %00000001 ; Set odd parity
B9600:  EQU !52 ; Baud rate = 9600
SC0BDH: EQU $C0 ; Baud rate register
SC0CR1: EQU $C2 ; Control register 1
SC0CR2: EQU $C3 ; Control register 2
SC0SR1: EQU $C4 ; Status register
SC0DRL: EQU $C7 ; Data register

;***************************************************************
; Subroutine sci_char_ready
; Check the RDRF flag
; If a character is ready, returns with C=1
; the character in the A register, and the
; status information in the B register.
; Otherwise, C=0 and the A and B regs are
; unchanged
; Inputs:  None
; Outputs: A = character, Carry bit T or F
;           B = status information
; Reg Mod:  A, CCR
sci_char_ready:
    clc                 ; Clear carry
    IF RDRF is set
        brclr SC0SR1,RDRF,exit
    THEN the character is there
        ldaa SC0DRL ; Get the data
        ldab SC0SR1 ; Get the status
        sec ; Set the carry
    ENDIF
exit:       rts
Parallel & Serial Peripheral Interface

PARALLEL COMMUNICATIONS

Port A
clk

Port B
clk

0x68 0x65 0x6c 0x6f
h  e  l  l  o

Need 10 wires to transmit 8 bits of data

SERIAL COMMUNICATIONS

serial out
clk

serial in
clk

0 1 1 0 1 1 0 0 0 0

0x68 = ‘h’

Need 3 wires to transmit 1 bit at a time

Dr. Tao Li
Synchronous Serial Communications

Dr. Tao Li
SPI Layout

- SHIFT REGISTER
- BUFFER REGISTER
- SHIFT CLOCK
- MOSI
- MISO
- SCK
- SELECT
- M68HC12 MASTER
- SS
- M68HC12 SLAVE
Synchronous Serial Peripheral Interface

• Sends high-speed serial data to peripherals, other SPI-equipped MCUs or DSPs

• Up to 4Mb/s, LSB or MSB sent first, normal or open-drain output for wired-OR

• Master/slave arrangement, master provides clock (SCK) to shift data in and out

• Data xfer’d out of each shift reg. simult. so master sends to and receives data from slave
Synchronous Serial Peripheral Interface

• Transmitted data is single-buffered; s/w must await last bit shifted out before writing new; SPIF (SPI xfer complete flag) available for polling and interrupts

• Received data is buffered, so program has one char. time to read data before next arrives

• Feature for master to select slave (e.g. to use for its CS) via SS* (slave select) output signal
SPI initialization

- SPI interrupt enable (SPIE bit, 1 = enabled)
- SPI system enable (SPE bit, 1 = enabled)
- Wired-OR mode (SWOM bit, 1 = open-drain outputs)
- SPI master/slave mode select (MSTR bit, where 0 = slave and 1 = master)
SPI initialization

- Slave select output enable (SSOE bit, 1 = enabled assuming DDRS7 = 1)

- SPI LSB first enable (LSBF bit, 0 = MSB first, 1 = LSB first)

- Serial pin control (SPC0 flag; 0 = normal 2-wire mode, 1 = bidir. 1-wire mode)
SPI Master and Slave Modes

- One master unit and one or more slave units via MSTR bit on each

- Normal 2-wire mode (4 pins used on each unit, 2 of it for data)
• Bi-directional mode (3 pins used in each, only 1 for data)
  – allows unused SPI bits (PS4 on master, PS5 on slave(s)) as GP I/O signals
SPI Data Rate and Clock Formats

• Data rate set by SPR2:SPR0 prescale bits as division of bus clock from $\div 2$, $\div 4$, $\ldots$, $\div 256$
  
  $\Rightarrow$ for bus clock = 4MHz, rates from 2MHz down to 15.625 kHz
  $\Rightarrow$ for bus clock = 8MHz, rates from 4MHz down to 31.25 kHz

• SPI clock polarity select (CPOL) bit, 0 = SCK low when not shifting data, 1 = high

• SPI clock phase select (CPHA) bit determines rising/falling in concert with CPOL
SPI Clock Phases
SPI Flags and Interrupts

• All flags reset by reading SP0SR register where flag resides followed by R or W access to SPI data register SP0DR

• **SPI interrupt request flag (SPIF) set** at end of SPI xfer; if SPIE set then SPI interrupt occurs

• **Write collision error flag (WCOL) set** if SP0DR written while data xfer in place; no interrupt with this flag

• **Mode error flag (MODF) set** if SS* pulled low while SPI in master mode (meaning some other SPI device trying to act as master and data collision may occur); if SPIE enabled then SPI interrupt occurs

• Since SPIF and MODF share same interrupt, ISR must poll to determine source