Recap

Serial communication; what does the frame look like
SCI shift register versus SCI data register
How RDRF is set; how RDRF is cleared
How TDRE is set; how TDRE is cleared

Overview

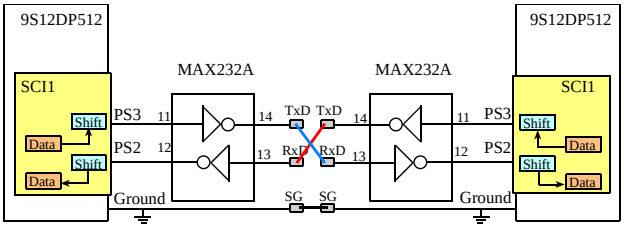
Baud rate vs bandwidth
Latency vs real time
Synchronization between computers
SCI interrupts

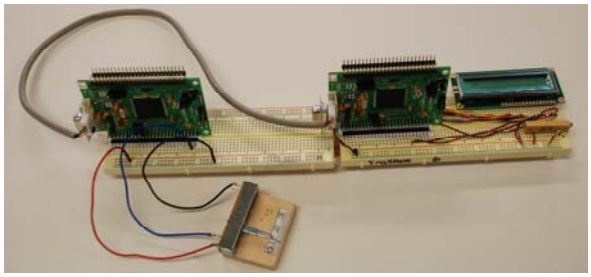
Classification

Simplex: data flows in one direction
Half-duplex: data flows in both ways, one way at a time
Full-duplex: data can flow both ways at the same time

Lab 8

hardware allows for full duplex communication
it is used in a simplex manner





Things to remember

There are two data registers at the same address SCI1DRL
- The transmit SCI1DRL is write only
- The receive SCI1DRL is read only

There are two shift registers
- The transmit shift register is connected to PS3 output
- The receive shift register is connected to PS2 input

Frames have 1 start, 8 data and 1 stop bit

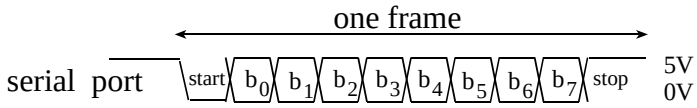


Figure 8.1. A 10-bit serial data frame(with M=0).

Review of how SCI works

To transmit, the software (busy-wait synchronization)
- Waits until TDRE is 1 (meaning data register is empty)
- Writes data to SCI1DRL

To transmit, the hardware
- Moves 8-bit data from data register to shift register
- Adds start bit at front (0)
- Puts stop bit (1) at end
- Shifts the 10-bit frame out PS3 at the baud rate

To receive, the software (busy-wait synchronization)
- Waits until RDRF is 1 (meaning data register has data)
- Reads data from SCI1DRL

To receive, the hardware

Waits for a 1 to 0 edge defining the start bit
Shifts the 10-bit frame in from PS2 at the baud rate
Checks to make sure there are proper start/stop bits
Moves 8-bit data from shift register to data register

Performance issues (Bandwidth, latency, noise immunity)

1) Bandwidth (information/sec)

bit time is the time per bit

baud rate is the total number of bits per time transmitted
information (data),
plus overhead (start, stop, parity)

$$\text{baud rate} = \frac{1}{\text{bit time}}$$

bandwidth is the information bits per time transmitted
same as **bit rate** and **throughput**

$$\text{Bandwidth} = \frac{\text{number of information bits/frame}}{\text{total number of bits/frame}} \bullet \text{baud rate}$$

Performance issue

2) Interface latency
Time from request to time of service
Real-time systems have bounded latency
E.g., time from RDRF=1 to software read SCI1DRL
E.g., time from TDRE=1 to software write SCI1DRL

Consider this case
SCI receiver is initially idle
Characters “1” “2” and “3” arrive one right after the other
Software does not notice RDRF
When is data lost?

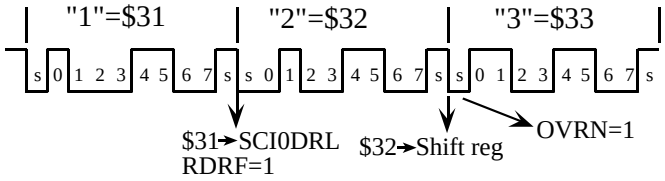


Figure 8.5. Three receive data frames result in an overrun (OR) error.

Overrun error (lost data) if
One frame is received, data put in data register
Second frame is received, this frame is in shift register
Software does not notice RDRF is set
Third frame begins (no place to put it)

Receiving serial data requires a real-time interface
Software must read data within 10 bit times of RDRF

Performance issue

3) Noise immunity
Probability data will be transmitted without error
We could add parity to detect errors
11-bit frame (M=1, PE=1, PT=0)
Even parity means #1’s is an even number
On transmission, parity = eor(bit_n), n=0-7
On reception, error = eor(bit_n, parity)
Cell phones use error correcting codes to fix mistakes

Solution to real-time system
Interrupt on RDRF
Never disable interrupts for longer than 10 bit times

8.2.4. 9S12 SCI Details

address	ms b														lsb	Name
$00D0	-	-	-	12	11	10	9	8	7	6	5	4	3	2	1	0	SCI1BD

Address	Bit 7	6	5	4	3	2	1	Bit 0	Name
$00D2	LOOPS	SWAI	RSRC	M	WAKE	ILT	PE	PT	SCI1CR1
$00D3	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCI1CR2
$00D4	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF	SCI1SR1
$00D5	0	0	0	0	0	BRK13	TXDIR	RAF	SCI1SR2
$00D6	R8	T8	0	0	0	0	0	0	SCI1DRH
$00D7	R7T7	R6T6	R5T5	R4T4	R3T3	R2T2	R1T1	R0T0	SCI1DRL

Similar to Table 8.4. 9S12 SCI ports (SCI1).

How to set up RDRF to interrupt

- Configure SCI
  - Enable TE RE
  - Arm RIE (interrupt on RDRF)
  - Set the baud rate
  - No parity, M=0
- Initialize data structure
  - Lab 7 used a mailbox
  - Lab 8 will use a FIFO
- Interrupt vector for SCI1
- Enable interrupts after all devices are initialized

RDRF interrupt service routine

- Invoked when (RDRF=1, RIE=1, I=0)
- Reads new input from SCI1DRL, puts into FIFO
- Latency equals maximum time system runs with I=1

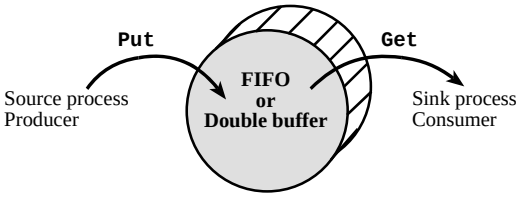
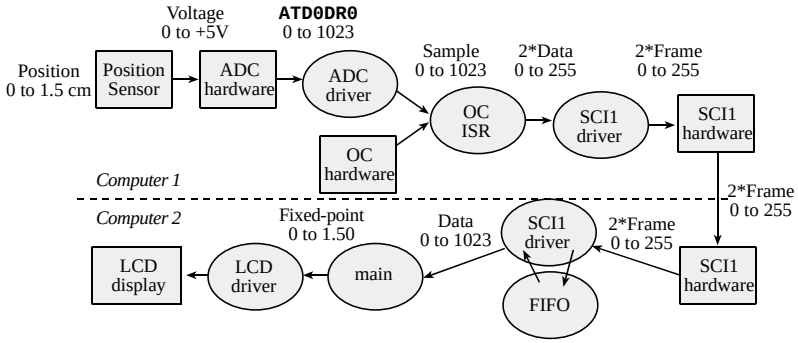


Figure 12.4. FIFO queues and double buffers can be used to pass data from a producer to a consumer.



Lab 8, Figure 8.3. Data flows from the sensor through the two microcontrollers to the LCD. The output compare timer is used to trigger the real-time sampling. Use the special serial cable to connect the two SCI1 ports.

Lab 8

- Busy-wait synchronization for transmission
- Interrupt synchronization for reception

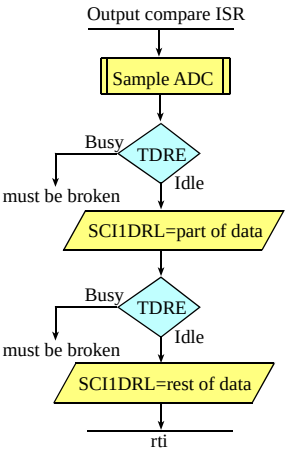
Flowcharts for the two systems in Lab 8

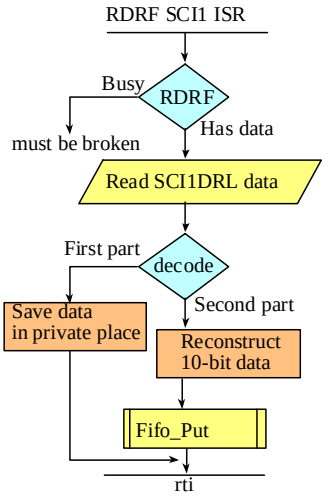
Fall 2010

Transmit 8-bit data as one 8-bit SCI frame

Spring 2011

Encode/decode 10-bit data into/out of two 8-bit SCI frames



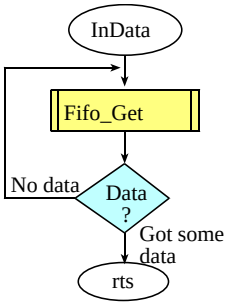


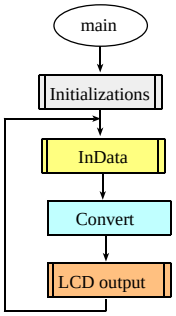
Transmitter ISR establishes real-time data acquisition

1) acknowledge the output compare interrupt
2) specify the time for the next interrupt, every 100ms
3) toggle PP7, heartbeat
4) sample the 10-bit ADC
5) send the 10-bit data to the other computer as two frames
6) increment a `Counter`, used as debugging monitor
7) return from interrupt

Receiver ISR

1) acknowledge (read status, read data)
2) read the data received from SCI1DRL
Look at data (according to your method of encoding/decoding) if first part
3a) store first part in a private global variable
If it is the second part
3b) toggle PP7 (change from 0 to 1, or from 1 to 0)
4b) reconstruct 10-bit data and put 10-bit data into the FIFO queue
5b) increment a global error count if the FIFO is full
If full don't loop back, just throw data away
6b) increment a `Counter`, used as debugging monitor
7) return from interrupt





`SCI1_InData`

1) Call Fifo_Get over and over until data is returned
2) Return the 16-bit data

Receiver Main

1) initialize PLL, FIFO, LCD, PP7, SCI, and enable interrupts
2) calls `SCI1_InData`, which will wait until new data arrives
3) convert sample to fixed-point (same as Lab 7)
4) output the result as a fixed-point number (same as Lab 7) with units
5) repeat steps 2,3,4 over and over

The bottom line

Baud rate, bandwidth, latency, real time
SCI interrupts