

COSC 3215 Embedded Systems Laboratory

Lab7 – Infra Red Data Link

Introduction

Throughout the first part of the course we have treated the HCS12 as a single-chip microcontroller, where the RAM, program, and I/O ports are internal to the microcontroller chip itself. The HCS12 has impressive I/O capabilities but sometimes we may need to map external devices into the memory map of the controller (for instance a larger RAM). With an external memory interface we can access external hardware as a memory location where data is accessed via simple read/write operations making things more convenient. This type of access is possible with the HCS12 in 'expanded mode' and is referred to as memory mapped IO.

Operating the HCS12 in one of the expanded modes changes the memory map. (Refer to Figure 8 "MC9S12DP256 Memory Map after Reset" in the MC9S12DP256.pdf document for a description on how the memory map is affected). The designer loses access to a variety of internal resources as well as the peripherals that are associated with the pins used for the external bus. Thus, the decision to use expanded modes of the HCS12 should be considered carefully. It is often possible to control an external hardware peripheral through an I/O port. However, in this lab we will use expanded mode to interface with an external peripheral to gain experience with memory mapped I/O.

Objective

In this lab you will design a Slow Infra Red (SIR) data link following the IrDA 1.0 specifications. The goal is to use this interface to transmit and receive messages from/to another group in your lab session. The messages will be typed and displayed on the console. You will combine both the DRAGON12 and the Altera (Verilog) in your design. Figure 7.1 through 7.3 show the IrDA 1.0 method for transmit and receive data encoding.

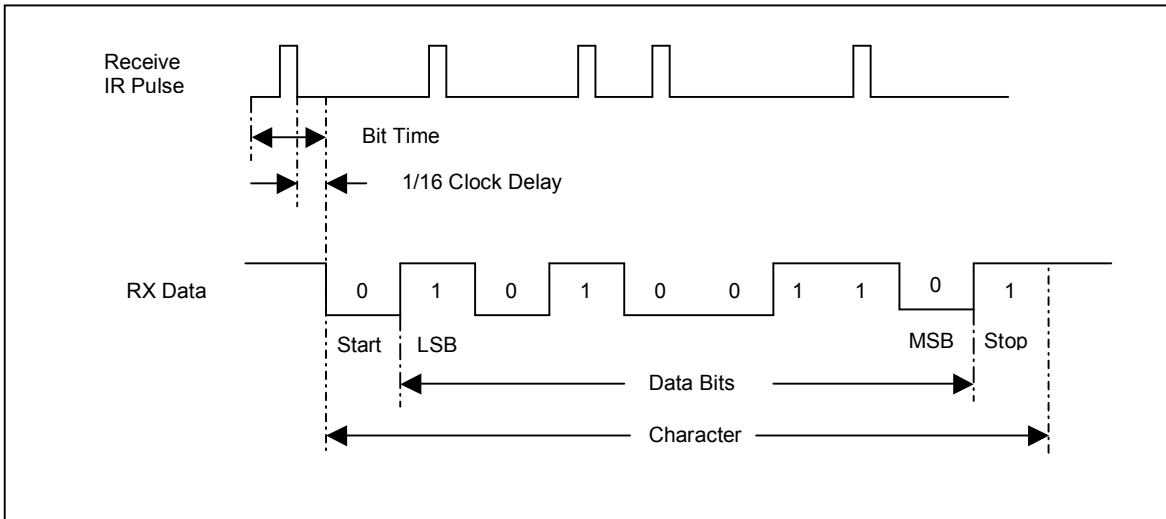


Figure 7.1 Standard Infrared Receive Data Encoding

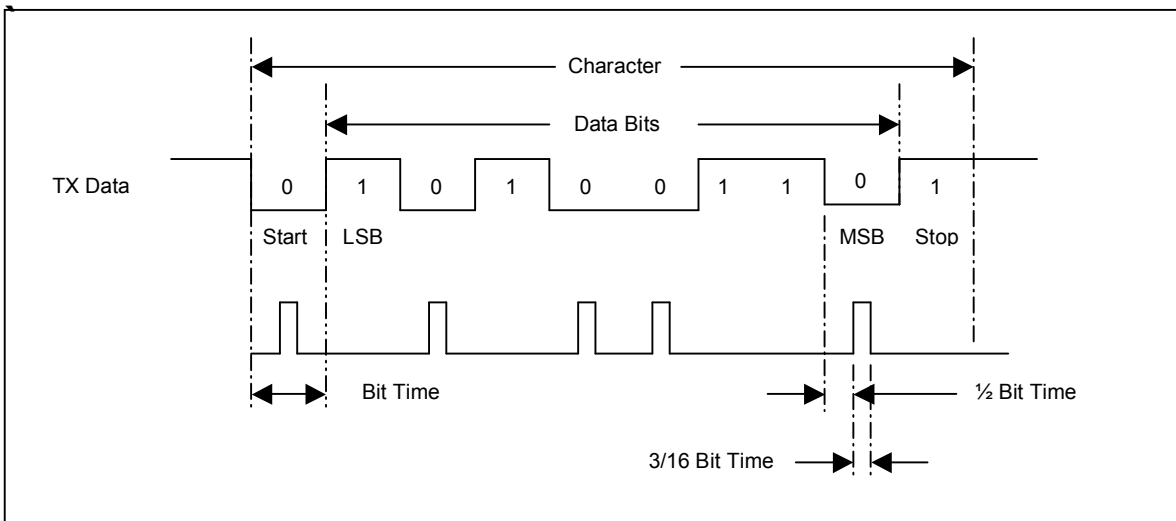


Figure 7.2 IrDA Infrared Transmit Data Encoding

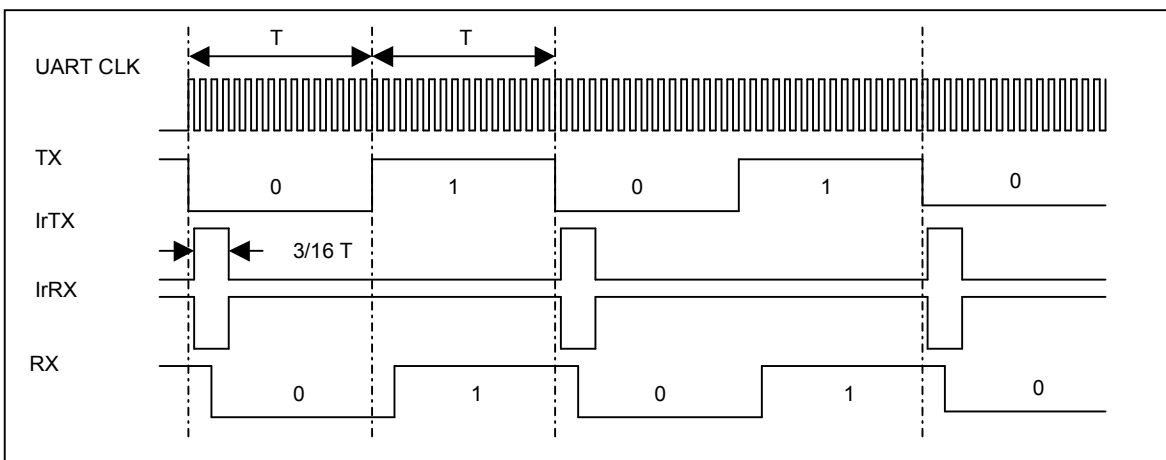


Figure 7.3 IrDA Transceiver Transmit & Receive Waveforms

Reference Reading

DRAGON12 Schematics
Dragon12_acc.pdf
Dragon12_1.pdf
Dragon12_2.pdf
Dragon12_3.pdf
Dragon12_4.pdf

MC9S12DP256B.pdf
MC9S12DP256 Advanced information

AN2287/D
HCS12 External Bus Design

AN2408/D
Examples of HCS12 External Bus Design

Design Specifications

The system can be broken down into three parts, 1) the microcontroller based RS232 serial communications interface with the console, coupled with data handling and buffering to the IrDA transceiver 2) the HCS12 expanded narrow mode hardware interface, and 3) the Altera (Verilog-based) IrDA communications peripheral that interfaces with the other IrDA transceiver. The transceiver should be designed to either receive data from or transmit data to another group while in an idle state. Once a process as begun reception or transmission then the alternate process is to be ignored. Meaning that if you are transmitting then ignore any reception and if you are receiving do not transmit. The purpose of this is to ignore the higher level protocols involved with IrDA SIR communications and put more emphasis at the hardware level of the IrDA data link.

The Console Interface

1. The DRAGON12 will handle the RS232 serial communications with the console using the Debug12 PutChar and GetChar routines.
2. Dipswitch SW1 is the source for the IrDA transceiver configuration and control settings, i.e. receiver/transmitter enable, baud rate selection, and receiver/transmitter interrupt enable. The IrDA transmission and receptions must be handled using interrupts.
3. Data must not be lost and **must** be buffered until the IrDA transceiver/console is ready for the next datum.

The Expanded Narrow Mode Hardware Interface, the Altera (Verilog)

1. The DRAGON12/HCS12 will operate in Expanded Narrow Mode.
2. The Altera based transceiver will conform to the external hardware interface requirements of the HCS12 when operating in Expanded Narrow Mode.
3. The interrupt source for the IrDA Transceiver will be the XRQ interrupt.
4. Set the base address of the IrDA transceiver to \$4024.

The IrDA Communications Interface, the Altera (Verilog)

1. The Altera will handle infrared data transmission/reception following the IrDA 1.0 transmit and receive data encoding methods shown in Figure 7.1.
2. IrDA communications shall be configured with 1-Start bit, 1-Stop bit, 8-Data bits, and no parity. The baud rate is programmable at either 19.2kbps, or 38.4kbps.

Pre Lab

1. Read the **Bus Control and Input/Output, Operating Mode and Resource Mapping** sections of the MC9S12256DP.pdf manual.
2. Develop your HCS12 communications interfaces (in either assembler or C) following the design specifications.
3. Develop your Expanded Narrow Mode hardware interface in Verilog targeting the Flex10 EPF10K70RC240-4 FPGA chip. Refer to the timing diagram found in Figure 12 “**Example Timing Analysis**” in the “**AN2408/D Examples of HCS12 External Bus Design**” application note. This document is available from www.freescale.com or from the course website. Please be aware the ECS/XCS signal is meaning less in Expanded Narrow Mode.
4. Develop your IrDA Transceiver in Verilog targeting the Flex10 EPF10K70RC240-4 FPGA chip. The details of the interface (register maps and control signals) are outlined in Appendix A. The pin connections for the DRAGON12 and IrDA transceiver are listed in Table 7.1

Hints

Figure 7.4, found at the end of this document shows a block diagram of the hardware interface, and IrDA transceiver. This diagram is not complete; some details have been intentionally left out. The implementation of these details and those of the internal workings of the transceiver are left to you.

This is a perfect application for the use of the logic analyzer, therefore it is suggested that you make use of it.

Note to meet the memory mapped I/O objective Debug12 is operating in the “Jump to EEPROM” mode where upon reset the Debug12’s startup code jumps directly to \$0400 without performing any initialization of the CPU registers. A boot-loader program has been placed at location \$0400 that configures the registers associated with operating the HCS12 in one of the expanded modes as well as setting the internal phase-lock-loop to give the required 24MHz clock frequency. The boot-loader also calls Debug12’s main function. This allows you to load/run your program from RAM, as you have done throughout this course. This also gives you access to the PutChar and GetChar routines thus hopefully simplifying the console serial communication portion of this lab. What you lose is the ability to set break points, examine memory locations and trace through your program.. This will make it difficult to debug your program. It may be helpful to toggle bits on an output port at specific points in your program, especially inside interrupt service routines. This will at least tell you that you have made it to this point in your code. PORTP bits 0 to 3 are available for this purpose.

The IRQ interrupt must be configured before it can be used. There is information on the IRQ interrupt through out the MC9S12256DP.pdf document.

Your Verilog design may simulate properly based on the timing diagram you use to perform the simulation. If this timing diagram is incorrect then you could waste valuable time trying to figure out why your design does not work when you are convinced that it should work. Avoid this frustration by bringing signals that would typically be internal to your design out on the one or more of the many output pins at your disposal. These signals can be removed from your design once you are finished with them. Each Altera board comes with a users guide listing the output pins and there corresponding connection to the edge connector.

The DRAGON12 is essentially a repeater; any data received from the console is immediately transmitted via the IrDA transceiver. Note the console and IrDA transceiver baud rates are different therefore you will need to provide an appropriate method of data buffering.

Remember to check the “What’s new” section on the course website for new information pertaining to this lab exercise.

In Lab

Procedure

1. Adjust SW7 until the **EEPROM** LED is illuminated then reset the DRAGON12.
2. The DRAGON12 should respond with the normal Debug12 header and prompt.
3. Download your program onto the DRAGON12.
4. Program the Altera with your IrDA transceiver and hardware interface following the same procedure as you used in Labs 5 and 6.
5. Use the logic analyzer to measure a read and write cycle. From your timing measurements determine the following:
 - a) Access time on a read operation
 - b) The write setup time
 - c) Data setup time from measurement

Evaluation

To show that the lab has been completed successfully you will transmit a text file to another group. This file will then be transmitted back to you where you will do a comparison with the original file. If the files match then you have completed the lab.

Appendix A:

IrDA Transceiver Control/Status Register

Address Offset: Base Address + \$01

	7	6	5	4	3	2	1	0
R	RDRF	TC	0	0	0	BRSEL	0	0
W			SHDN	RIE	TCIE	BRSEL	REN	TEN
Reset:	0	0	0	0	0	0	0	0

Read: anytime

Write: anytime

RDRF — Receive Data Register Full Flag

RDRF is set when the data in the receive data register is full. Clear RDRF by reading from the transceiver's data register.

1 = Receive data available in IrDA data register.

0 = Data is not available in IrDA data register.

TC — Transmit Complete Flag

TC is set low when there is a transmission in progress. TC is set high when the TDRE flag is set and no data is being transmitted. Clear TC by writing to the transceiver's data register.

1 = No transmission in progress

0 = Transmission in progress

RIE — Receiver Full Interrupt Enable Bit

RIE enables the receive data register full flag, RDRF to generate interrupt requests.

1 — RDRF interrupt requests enabled

0 — RDRF interrupt requests disabled

TCIE — Transmission Complete Interrupt Enable Bit

TCIE enables the transmit buffer empty flag, TC, to generate interrupt requests.

- 1 — TC interrupt requests enabled
- 0 — TC interrupt requests disabled

BRSEL — Baud Rate Selection Bit.

BRSEL selects a baud rate of either 19.2kbps or 38.4kbps.

- 1 — sets baud rate to 38.4kbps
- 0 — sets baud rate to 19.2kbps

SHDN — IrDA Transceiver Shutdown control bit.

SHDN activates or shuts down the transceiver. SHDN takes effect after the completion of the current transmit or receive operation.

- 1 — Complete shutdown of the transceiver.
- 0 — Enables the transceiver.

IrDA Transceiver Data Register

Address Offset: Base Address + \$00

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset:	0	0	0	0	0	0	0	0

Read: anytime; reading accesses IrDA receive data register

Write: anytime; writing accesses IrDA transmit data register

R7-R0 – Received bits seven through zero for 8-bit data formats

T7-T0 – Transmit bits seven through zero for 8-bit data formats

Appendix B:

Table 1 Signal Names & Connections

Signal Description	DRAGON12	Altera			
		FLEX_EXPAN_A	Flex10K	FLEX_EXPAN_B	Flex10K
Addr[15]	PA07	56	101	--	--
Addr[14]	PA06	55	100	--	--
Addr[13]	PA05	54	99	--	--
Addr[12]	PA04	53	98	--	--
Addr[11]	PA03	52	97	--	--
Addr[10]	PA02	51	95	--	--
Addr[09]	PA01	50	94	--	--
Addr[08]	PA00	49	88	--	--
Addr[07]	PB07	41	79	--	--
Addr[06]	PB06	42	80	--	--
Addr[05]	PB05	43	81	--	--
Addr[04]	PB04	44	82	--	--
Addr[03]	PB03	45	83	--	--
Addr[02]	PB02	46	84	--	--
Addr[01]	PB01	47	86	--	--
Addr[00]	PB00	48	87	--	--
ECLK	PE04	40	78	--	--
RW	PE02	39	76	--	--
IRQ	PE01	38	75	--	--
IrRX	--	--	--	23	118
IrTX	--	--	--	26	126
SHDN	--	--	--	25	120
VCC	--	59	--	--	59
GND	--	60	--	--	60
25MHz	--	--	91	--	--