Lab Time_____ Name_____

Lab 7. Serial Communication via UART(C)

Preparation

You will need a LaunchPad and a laptop/computer with Keil uVision5 installed. Download the Lab7 starter project from Canvas. You will also need to download PuTTY or a similar tool.

Starter project Lab7

Purpose

The purpose of this lab is for you to gain experience using the UART serial communications protocol.

Introduction

In this experiment, you will be modifying a given source project in C for a clock. Most of the functionality has already been provided. You will need to add some of the UART functionality. When completed, the program will ask you to select a mode of operation, either start time at 0 or enter a start time in seconds. The program will then update the time every second. The time will be displayed in a PuTTY terminal.

Program Operation Overview

The fully functioning program will operate as follows: first the program will ask you to press '0' to start the clock at time 0 or '1' to enter a specified start time in seconds. The function that reads your specified time is not very robust. Pressing any key other than a number (including backspace) will result in undefined behavior. The program uses the global variable g_seconds to track the number of seconds that have elapsed. This value will be incremented every second via SysTick interrupts. The program should then output the time in the format hh:mm:ss updating every second.

Resources

PuTTY – An open source program that will allow serial communication between your board and your computer. You can download the installer file or a copy of the executable from the following link: https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html. You will need to configure your connection when you first launch PuTTY. Choose Serial as your connection type. Next, select your COM port and a speed of 9600. You can determine the COM port by checking your Device Manager to see which one your board is connected to. A terminal window will be launched after clicking "Open". This will be the interface through which you can send and receive UART communications with the board.

ASCII – A type of character encoding in which a numerical value is used to represent a specific text character. For example, the character '1' actually has a numerical decimal value of 49. You may find it useful to use an ascii table for the last part of this lab. You can find one at this link (http://www.asciitable.com/) or you can use the one at the end of this handout.

UART – There are a number of materials available under the UART section of our main course page on Canvas. UART.pdf contains slides describing its operation. UART Notes.pdf provides a description of the steps required to configure UART. Lastly, the video "C11 3 UART code" provides a walkthrough of writing the code required to use UART.

Procedure

1. Begin by running the provided software project. Download the project, unzip the folder, and open the Keil uVision project file. Compile the project and load it on to the Tiva Launchpad Development board. Now open the device manager and determine the COM port associated with the board (Stellaris Virtual Serial Port). Launch PuTTY and open a serial connection with the COM port for your board and 9600 Baud.

   You now need to complete the InitConsole() function to initialize UART0. If done correctly, the PuTTY terminal should display a message stating that you have completed the first part of the lab. Demonstrate this to the TA and explain how you determined what value to use on each line.

Lab Time_____ Name_____

Signature_____ Date_____

2. Next, the program will give you some options and allow you to choose the one you want. You need to complete the function myGetChar() for the program to be able to correctly record your input. When called, myGetChar() should block until any key is pressed. It will then return the ASCII value corresponding to that key. Your program will advance when you press either '0' or '1'. It will respond with a warning message if any other key is pressed. The program will notify you that you have completed Part 2 after you press '0' or '1'. Demonstrate this to the TA and explain how your code works.

Signature_____ Date_____

3. Lastly, you will change the PrintTime() function so that the displayed time is continuously overwritten rather than displayed on a new line. The time is tracked in seconds by the global variable g_seconds. The time should be displayed in the format hh:mm:ss (assuming a 24 hour clock). Rather than using one of the provided UART STDIO functions such as UARTprintf(), you should complete myPutChar(). You will need to call myPutChar() multiple times inside PrintTime() since it can only output a single char at a time. Demonstrate your program's correct functionality to the TA and explain how your code works.

Signature_____ Date_____

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com