4

Designing Your Own Soft Modules

Objectives

- Learn how to create circuit schematics with OrCAD
- Learn how to export a circuit from OrCAD as an EDIF netlist.
- Learn how to import an EDIF netlist into the FastChip library as a new soft module.
- Learn how to use the imported module in CSoC designs.

Why Design Your Own Soft Modules?

By this time you have created designs using soft modules and the 8032 MCU alone and in combination. The keyboard interfaces in the previous chapter required you to connect multiple soft modules together and then interface it to the MCU. Two problems with this approach are:

- The soft modules are interconnected only through the names of the signal nets and this makes it difficult to visualize the flow of information between the modules.
- A subgroup of interconnected soft modules can't be transferred from one FastChip project to another so design re-use is difficult.

This chapter will show how these problems are solved by using OrCAD Capture. Capture lets you draw your circuitry as an easily-understood schematic built from standard logic symbols with interconnecting wires and buses. You can export your circuit as a netlist file in EDIF format and then import the netlist into the library of any FastChip project. The imported module acts just like the standard soft modules in the FastChip library. Simply instantiating the imported module will bring all the functionality of your circuit into your FastChip project. The rest of this chapter will show an example of how to design a new soft module with OrCAD Capture and use it in a FastChip project.

Design 4.1 - PS/2 Keyboard Interface Module

In Chapter 3 you built an interface from soft modules that received a serial bit stream from a keyboard and interrupted the 8032 MCU when a complete scan code was available. Now you will rebuild that interface as a single soft module using OrCAD Capture.

The schematic for the keyboard interface circuit is repeated in Figure 19. Falling edges of the **ps2_clock** signal strobe keyboard scan code bits from **ps2_data** into the **ps2_data_sreg** shift register. Each rising edge of **ps2_clock** sets the **rcv_active** flip-flop which indicates the receiver circuit is gathering scan code bits. The **bit_timer** counter is also cleared whenever **ps2_clock** is low. Once **ps2_clock** stays high at the end of the scan code transmission, the 25 MHz BusClock will have sufficient time to increment the counter until bit **bit_timer[11]** goes high. Once **rcv_active** and **bit_time[11]** are both set, this drives the **rcv_int_comb** signal high which 1) clears the **rcv_active** flip-flop indicating the receiver is no longer active, and 2) sets the **rcv_int** flip-flop that strobes the scan code from the shift register into the **rcvData** register and sends an interrupt (**INTR0**) to the 8032 MCU. When the MCU responds to this interrupt, it reads the scan code from the logical address of the **rcvData** register. The presence of the **rcvData** register address on the CSI address bus triggers a selector. The **RdSel** signal goes high and this gates the scan code onto the CSI data bus while simultaneously clearing the interrupt from the **rcv_int** flip-flop.



Figure 19: Schematic of a PS/2 keyboard interface circuit that uses interrupts.

Starting the Design

You can start this design by loading the *Chap31* design into FastChip. Then delete all the soft modules in the keyboard interface and save the project under the name *Chap41*.

📈 Triscend FastChip: Cl	hap41 (Target Device: TE505S16-25L)	_ 🗆 ×
📈 File View Tools Help		
Project Library Import	EDIF I/O Editor Generate Bind Download Debug Log Stop	1
Triscend Library ? X	Dedicated Resources ?	
 Archive Displays Logic Modules Peripherals CSI Bus I/O Memory Communications 	Configurable System Logic ?	atchdog
	The American Ame	
	ps2_data ps2_clock LED	
Resource Estimate	2	
LCSL Cells: 7/512	1 % 1/0 Pins: 9/60 ∎15 % ⊕CSI Selectors: 1/32 3 % Performance: 25 MHz	2
✔Ready.		

Starting a Project with OrCAD Capture

Now that you have the FastChip project started, create a folder named orcad in the Chap41 FastChip project folder. Your OrCAD schematic files will be stored there. Then

click on the common icon to start the OrCAD Capture software.

In the **OrCAD Capture** window, select File⇒New⇒Project... to start a new Capture project.



In the **New Project** window that appears, click on the Schematic radio button since your design will be done as a schematic. In the Name box, type ps2port as the name of your keyboard interface module. Then use the Browse button to select the orcad folder under the Chap41 project folder as the location for the Capture project files. Then click on OK to accept these settings.

Ne w Project			X									
Name		ОК										
ps2port		Cancel										
Create a New Project Using	a New Project Using											
C Analog or Mixed-Signal Circuit Wizard												
PC Board Wizard Programmable Logic Wizard	Tip for New Schematic' fastest way schematic p	Users Wizard is the to create blank project.										
Schematic												
L <u>o</u> cation c:\CSoC_Examples\Chap41\orcad		Browse										

Now the OrCAD Capture window will contain two subwindows:

ps2port.opj: This window shows the organization of the entities in this project. The File and Hierarchy tabs let you select whether you want to view the organization of the project files or the circuitry components.

SCHEMATIC1:PAGE1: You will draw your schematic in this window.



Now you need a library of circuit elements that you can interconnect to build your circuit. Triscend supplies several libraries of components that are specially designed for the CSL matrix of the CSoC. You should only use components found in the Triscend libraries. To add these libraries to your OrCAD project, right-click on the Library folder in the **ps2port.opj** window as shown below. Then select Add File in the pop-up menu that appears.



In the **Add Files to Project Folder - Library** window, steer your way through the folder hierarchy shown below to find the Triscend schematic libraries.

Add File t	o Project Folder - Library				? ×
Look <u>i</u> n:	🔁 Orcad	•	🖻 💆		
DEMOR	My Computer				
📓 Triscer	(C.)				
	Program Files Triscend				
	EastChip_1999				
File <u>n</u> ame:	Schematic				en
Files of <u>t</u> yp	;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;	n)	-	Can	cel
File <u>n</u> ame: Files of <u>t</u> yp	Chematic Crcad CC:) Removable Disk (E:)	Å		Ope Cano	en cel

Select all three of the Triscend schematic libraries in the folder and click on Open.

Add File to Pro	ject Folder - Library	? ×
Look <u>i</u> n: 🔁 🤇	Drcad 🔽 🔁 🖻 🗐	
DEMORGAN TRISCEND. TriscendMa	NS.OLB DLB cros.olb	
	$\Box_{\!S}$	
J		
File <u>n</u> ame:	"TriscendMacros.olb" "TRISCEND.OLB" "DEMOR	n
Files of type:	Capture Libraries (*.olb)	el

Once you click on Open, you may receive the message shown below. The Triscend libraries are stored in a format for version 7.2 of OrCAD Capture, but you have version 9.1 of Capture. You can elect to convert the Triscend libraries to version 9.1 or not. (I left all my Triscend libraries in version 7.2 format.)

OrCAD (Capture 🔀
<u>.</u>	c:\program files\triscend\fastchip_1999\data\libraries\schematic\orcad\triscendmacros.olb This file was created by Capture 7.2 data format. Would you like to save the file now in the current data format?
	<u>Y</u> es <u>N</u> o

Now the Triscend libraries should be listed under the Library folder in the **ps2port.opj** window.



Drawing a Schematic with OrCAD capture

With the Triscend libraries in place, you can start drawing your keyboard interface schematic. Click in the **SCHEMATIC1** window and the tools for entering a schematic appear along the right-edge of the **OrCAD Capture** window.



To start, you will use the part placement tool to place parts from the Triscend libraries into your schematic. Clicking on 🖻 brings up the **Place Part** window. You can highlight one or more of the entries in the Libraries area and the parts contained in the highlighted libraries are listed in the Part area. When you highlight one of the entries in the part list (a D flip-flop in this case), a small symbol for the part will appear. Once you find the part you want, click on OK.

Place Part		×
Part:	Graphic	ОК
	© <u>N</u> ormal	Cancel
		Add Library
	Packaging Parts per Pkg: 1	Remove Library
DHFS DMACTRL DMUX	Par <u>t</u>	Part <u>S</u> earch
DR7_0		Help
Lįbraries: DEMORGANS Design Cache TRISCEND TRISCENDMACROS	D Q C C C C C C C C C C C C C C C C C C	

Now the part symbol is attached to the cursor as you move it within the **SCHEMATIC1** window. Clicking with the mouse drops a copy of the part symbol into the window.

CrCAD Capture	×
<u>File E</u> dit <u>View</u> Place <u>Macro</u> <u>Accessories</u> <u>Options</u> <u>Window</u> <u>H</u> elp	
PC8 File *. Hierarchy *. Design Resources *. Library *. C\progr *. C\progr	
0 items selected Scale=100% X=0.90 Y=1.50	A

The part symbol remains attached to your cursor even after you click the mouse. This lets you add several independent copies of a part without having to go back to the **Place Part** window each time.



When you want to stop adding a particular part to your schematic, you can right-click with the mouse to make the following pop-up menu appear. Select End Mode and the part symbol will disappear from the cursor. You can also get the same effect by pressing the ESC key.



Each part that you drop into a schematic is given a default name (**DFF_1** and **DFF_2** in this example). You can give a part a more descriptive name by double-clicking on the part name. Then just type the part name you want in the Value field of the **Display Properties** window that appears.

Display Properties	×
Name: Part Reference V <u>a</u> lue: ps2_data_sreg0	Font Arial 5 (default) <u>Change</u> <u>U</u> se Default
Display Format © Do Not Display © Value Only	Color Default
C Name and Value C <u>N</u> ame Only	Rotation
C Both if Value Exists	C <u>9</u> 0* C <u>2</u> 70*

Once you add the parts to your schematic, you can rearrange them by clicking on the tool and then click-and-drag the symbol for the part you want to move. Drag it where you want and then release it. If you drag one part close to another, the pins on each part will be highlighted. This indicates a connection will be established if you drop the part at that location. This is a quick way to attach parts to build larger functions. For example, a byte-wide register can be built by concatenating eight D flip-flops together.



However, in most cases you will connect parts using the wiring tool. Click on the L tool and then click on a pin of one of the part symbols. As you move the mouse, a wire will extend from the pin. Click the mouse each time you want to change the direction of the wire.



Once your wire gets close enough, a pin will be highlighted. Clicking the mouse will terminate the wire on that pin and establish a net connection between the two parts.

	C)rC	AD	Ce	apt	ure	- [/	- (S	€CF	IEM,	AT	IC1 : I	PAG	E1)]															_	_ [/		×
	Ì	<u>F</u> ile	Э	<u>E</u> di	it	⊻ie	w J	Plac	e	<u>M</u> ac	cro	<u>A</u> cc	esso	ories	<u>0</u>	ptio	ns	₩i	ndo	W	<u>Η</u> ε	elp								- 1	8	×
Ť	2	B		e	∌	Х	Pe	ß	Ľ	20	×	DFF		-] [₽ 0	3	2	8	U?	\$1		۶ 🕅) IX			[]\$	쀁	?			
[c]										5		-					Т									4				-7		*
Ш																															Ľ	
Ш																															Ir	-
Ш																															ľ	N1
Ш									<u>п</u> .					- -																	ŀ	╡
Ш								΄	Ĩ.,	DF	F		۰ ·																		ŀ	+
D									F١	J - 2		CLE																			_	7
Ш										• /		<u> </u>) ne	2 4	lata		roc	à													1	PHR
Ш									0	· [1	. 🗖	ps	<u>~_</u> u	ato		ieg														1	GND
Ш																															1	
Ш																															1	
Ш								<u>_</u>		<u> </u>			_																		1	Ю Н
Ш									D			- <mark>-</mark>																			1	≪c
Ш										DF	F		`																			×
H.									F١	J - 2		CLE	2																			$\overline{}$
Ш										• /	T		ns.	2 d	lata	a . s	rec	i0													1	ź
Ш										· [1		10	~	i aire			,•													1	
																															1	0
Ц																															-1	5
H		_		_	_	_					_			-	40.4			e			0.00	0/	v	_0 0	0 5	/_1	EO		_	-	Ŀ	A
												o iter	ns s	eiec	, eo			5	cal	e=2	00	70	X	-0.9	U	r = 1	.50				//i_	

You can also attach a wire to a pin without terminating it at another pin. Just right-click while drawing the wire and select End Wire on the pop-up menu as shown below.

👫 OrCAD Capture - [/ - (SCHEMA	TIC1 : PAGE1)]	×
🛐 Eile Edit View Place Mac	ro <u>A</u> ccessories <u>O</u> ptions <u>W</u> indow <u>H</u> elp	₹ N
``````````````````````````````````````	DFF - B. B. Q. B. U? #1 17 @ IX II 19 %. ?	
		1
		1-1
		니니
		┛╧╢
		PHR
	ps2_data_sreg1	Q GND
		-
		ien
	End Wire	
Н	Ascend Hievarchy	음
	CLR Zoom In	$\geq$
· · · · · · · · · <u>-</u> · <u>-</u>	ps2_data_sreg Zoom Out	<u> </u>
	Go To	븩
		19
	<u> </u>	1
	0 items selected Scale=200% X=2.00 Y=1.60	A

You can assign an alias to any wire by clicking on the <u>state</u> tool. Type the name you want to assign to the net in the Alias box of the **Place Net Alias** window that appears. Then click OK.

Place Net Alias	×
Alias: s0	OK Cancel
Color Rotation	Help
Default         ▼         © <u>0</u> °         ©	<u>1</u> 80° <u>2</u> 70°
Font <u>Change</u> <u>Use Default</u> Arial 5 (default)	

The new name for the wire will be attached to the cursor. Then just click your mouse on the wire you want to name.

	0	)rC/	٨D	Cap	ture	- [/ -	(SC	НЕМА	TIC1 : I	PAGE1)	]												_ [	] ×
E	3	<u>F</u> ile	E	dit	⊻ie	w <u>P</u>	lace	<u>M</u> acı	ro <u>A</u> cc	essories	: <u>O</u> ptic	ons	<u>W</u> ind	ow	<u>H</u> elp	)						[	_ [2	₽ ×
ł	3	B	8	8	X	₽₽ I	8	<u>n</u> c	DFF	•	- 🕵	<u>a</u>  9		U?	‡I[	DRC [	di IX		E.N		리	8		
Ē				_				5										4						1
Ш																							1	극극
Ш																								
Ш																								
Ш							Ľ																	<u><u>N1</u></u>
Ш						• [===	10	DEE	- C															느
D								DFF															j L	1+
П							E	N /		2														
Ш								. [	<b>ا</b> م . ا	ps2_	data_s	sreg	1											- Que
Ш																								<u>-</u>
Ш																								<u> </u>
Ш																								
Ш									- 8	<b>.</b>		.												<u> </u> ⊚H
Ш						-	D		_ · C				<u> </u>											<u> «c</u>
Н								DFF					· .											×
Ш							E	N 📝		2														$\sim$
Ш								а <u>г</u>		ps2	data	sreg	0											×۷
Ш							· L			• •														
																								0
Н	4																						Ì	
쁜	• 1							_	0 iter	ne colo	cted		Sca	lo=2	00%	5	(=2 N	n v	=16	Λ			-	A
	<u> </u>								0 iter	ns sele	cted		Sca	le=2	00%	>	<=2.0	0 Y	=1.6	0				

This attaches the alias to the net.

	Or	CAI	D C	ap	ture	- [/ -	(SC	нем	ATI	C1 : P	AGE	1)]															_		×
	È	ile	<u>E</u> 0	dit	⊻iev	v <u>P</u>	ace	Ma	cro	<u>A</u> cce	ssorie	es	<u>O</u> ptio	ons	Wi	nda	W	<u>H</u> el	р								_	8	×
Ť	2   2	3   E	3	8	Ж	<b>Pa</b> (	8	<u>_</u>	2	DFF		-	₿.	ß	] ا	<u></u>	U?	‡1[	DRC	١Ņ	<u>EX</u> E		1 🗄	5	暭.	ę	1		
Ē								5				_					_					4	_		_	_	-	Ħ	
Ш																													
Ш																													2
Ш																													그
Ш							E	] [			1																		<u>N1</u>
Ш							D		-	· Q																			긴
Ш								DF	F		1.1																		+
D							E	N 📝		CLR	1																		1
Ш								, í		<u> </u>	ps2	da	ta	sre	<b>q</b> 1														PHR
Ш								. 1	•	. 💾																			GND
Ш																													::
Ш																													
Ш																													Г
Ш						<b>F</b>	Ь			. o					_s0														≪c
Ш							٢	DF	F	~																			×
H							-	N	~																				
Ш							E			ULK		1			-														$\overline{\mathbf{x}}$
Ш							· [	. I		. 📥	ps2	_da	ita_	sre	gu														÷-
Ш																													븻
																												<b>+</b>	읙
Ŀ																											F		-
										0 item	s sel	ect	ed		S	cal	e=2	00%	6	X=;	2.80	) Y	=0.2	20				1.	A

You can connect pins by attaching wires to each of them and assigning the same alias to each wire as shown below. This lets you avoid cluttering your schematic with many crossing wires.

E		OrC/	٩D	Cap	oture	- [4	/ - (8	SCI	HE	мат	1C1	: PA	GE1	)]															_		×
	1	<u>F</u> ile	E	dit	⊻ie	w	<u>P</u> la	се	M	acro	A	cces	sorie	s !	<u>O</u> pti	ons	$\underline{W}$	indo	W	<u>H</u> el	р									6	ı ×
	ð	<b> </b> 2		8	X	Ē		<u>,</u>	<u>_</u>	<u>_</u>	DF	F		-	<b>₿</b>	R	Q	<u>8</u>	U?	‡1[	DRC	Ø	T TX			5	Ľą	[9	2		
Ē				5				_		_	-	1			_				4						_			_	_		
																															-
							50							_	<b></b>																
							50				– C	)		< 0	2 -	-															
												- Df	-F																		-
D											E	EN 1	$\wedge$	CLI	R															_	
													Н		р	s2_	data	a_s	rég	1											PHR
																															GND
																															Ē
																				s0											
												, DF	=E		-					_											
ŀ	1												~																		
											Ľ		$\leftarrow$		R	à	d at		ró a	ó											믬
												<u>+</u> .	₫	· t	l b	sz_	gara	a_Si	reg	Ų.											$\geq$
																															싐
L																														-	님
Ļ	•		_	_		_		_	_				<u> </u>		_		_				-									]	듹
											0 i	tems	s sel	ecte	ed		S	ical	e=1	88%	6	X=	0.8	0 1	7=2	.30					브

You can place parts and connect them with wires, but you also need a way to pass signals in and out of your circuit if it is going to serve as a module in a larger design. The module I/O is performed by ports. To add a port, click on the root. The **Place Hierarchical Port** window will appear with a list of the various ports you can drop into your schematic. Highlight one of the ports (I'm partial to **PORTRIGHT** but it really doesn't make much difference) and type a name for the port in the Name field. Then click on OK.

Place Hierarchical Port		×
Symbol: PORTRIGHT-R PORTLEFT-R/CAPSYM PORTNO-L/CAPSYM PORTNO-R/CAPSYM PORTRIGHT-L/CAPSYM PORTRIGHT-R/CAPSYM Libraries:	PORTRIGHT-R	OK Cancel Add Library Bemove Library Help
CAPSYM Design Cache	<u>N</u> ame: ps2_clock	

Now the port symbol will be attached to your cursor. Click the mouse to drop the port into the schematic.

E		OrC	AD	Cap	otur	e -	[/ ·	- (S	C⊦	IEN	A	ΓІС	1:	PA	GE	1)]												_ [	٦×
		<u>F</u> ile	e j	<u>E</u> dit	⊻i	ew	E	lac	е	M	acro	D,	<u>A</u> cc	ess	ori	es	<u>0</u>	ption	s <u>W</u> ir	idow	<u>H</u> e	elp						_ [	2 ×
	ħ	B		8		K	b	ß	4	2	2	D	FF			-	6	80		N U	? \$1		N D	XI II	i b	12	8		
ſ	2									5									<u> </u>						4				J.
																		-	3 6	]									
																	-	D			Q								14
																			DFF		÷.,								
																		E	N /	C	ĽR								
																				1	rh I	ps2	_dat	a_s	reg	1			
																					· ·			_					
																													PHR
																													GND
																						1				s0 [°]			
																	-	D			Q								
ŀ	1																		DFF		1								₩
																		E	N //	C	ĽR								≪c
																		. r	ц. р	].	h l	ps2	_dat	a_s	reg	ן כ			×
			- F			<u> </u>	<u> </u>			<u> </u>	<u> </u>			<u> </u>	<u> </u>	- 1													2
	1		-	ps2	_c	loc	k								D	Ð													
						÷	-				-			· ·			Ď	×											0
																		s .											12
		_				_	_	_	_	_	_	1	ite	me	ole	oct	be		ę,	ala-	2004	%	Y-1 '	70 \	/-2 A	n		₽	A
Ľ													ner	11 5	ele		sul		Jou	ale-	200.	~0	A-1.	.0 1	-2.4	0			//

Next you need to set the direction of the I/O port. Even though the symbol may appear to be that of an input, that doesn't really mean it is. You should always explicitly set the direction for the port. Double-click the port symbol to bring up the **Property Editor** window for this port instance. Select the direction of the port from the drop-down menu under the Type heading. In this case, the port brings the clock signal from the PS/2 port into the circuit, so the Input type is selected. I generally use only ports of type Input and Output when designing a module for a Triscend CSoC. If I need a bi-directional port I will add two ports - one for input and one for output.

🚯 OrCAD Capture - [Property Editor]	_ 🗆 🗙
🛐 Eile Edit View Place Macro Accessories Options Window Help	_ & ×
	₽ <b>?</b>
New Apply Display Delete Property Filter by: < All >	
Connected Net Name Source Library Source Symbo	l Type 🔺
1 SCHEMATIC1 : PAGE1 : ps2_clock ps2_clock ps2_clock C:\PROGRAM FIL PORTRIGHT-R	Input 🗾
	Input
	Bidirectional
	Onen Collecto
	Passive
	3 State 🖵
✓ ► A Schematic Nets A Pins A Title Blocks A Globals Ports A A	Open Emitter
1 item selected	Power

After setting the port direction, click on the Apply button and close the window. Now you can use wires to connect the port to the pins of other parts in your schematic.

	0	rCA	D C	ap	ture	- [/	- (S	C⊢	IE₩	IAT	IC1	l : F	PAG	ÈΕ	1)]									_	⊐×
	] [	<u>F</u> ile	<u>E</u> c	lit	⊻ie	w	<u>P</u> lac	e	<u>Μ</u> ε	acro	Ł	Acce	essi	orie	es	<u>0</u> p	otions	s <u>W</u> inc	low <u>H</u>	elp					5 ×
Ť	2 1	2) 		0	X	Ē	B	Ľ	2	2	DI	F			-	đ	N B		U? ‡	1 🗸		<u>I</u>	Ľą	?	
$\overline{\Gamma}$									5													4		_	
Ш																	FE	- 8		<b>1</b> -				1	- <del></del>
Ш																-	D		· Q	-E	]				
																	1	DFF		1					<u>N1</u>
Ľ																	E	N (	CLR						1
Н																		. 🗄	. 🗄	ps2	2_data	sreg1			+
Н																									1
Ш																								1	PHR
Ш																									GND
Н																				1		s	)		
Н																	יין	DFF							
H																			CLE						Ì⊚H
Ш																	Ц			ne	cteb (	erod			
Ш																	· [	] - 🔁	•	psz		siego			<b>H</b>
Ш																									$\geq$
Ш			'n	.2	cl	ock								-											<u>~</u>
H			P	32		UCK								~											늬
																									귀
Ŀ	(																							▶	
										0	ite	ms	sel	ec	tec	1		Scale	=200%	X	=2.10 Y	=2.00			

This completes my short introduction to the tools for entering a schematic. OrCAD Capture has many features and shortcuts that I haven't covered, so you should read the documentation if you want to learn more.

The one-page Capture schematic for the keyboard interface circuit is shown below. It follows the structure and naming of the circuit in Figure 19. Each section of the circuit will be shown in more detail.



A 10-bit shift register gathers the scan code bits from the **ps2_data** input port as pulses are applied to the **ps2_clock** port. The eight scan code bits are clocked into the **rcvData** register when the **rcv_int** flip-flop is set.



The output of the **rcvData** register is gated onto the CSI data bus through the **DR7_OE_1** component when its **EN** input is driven high.



The EN input of the DR7_OE_1 component is driven by the RDSEL output of the SELECTOR_1 component. In this case, SELECTOR_1 responds to a single address in the 8032 SFR address space when this address appears on the CSI address bus. This is expressed in the schematic by attaching the SIZE1 and P8032 SFR components to the SELECTOR component. The actual SFR address is not specified. Instead, a SYMBOLIC component is attached to the SELECTOR through a wire with the alias rcvData. The FastChip software will replace the symbolic address with a logical address in the SFR space of the 8032. The combination of the DR7_OE, SELECTOR, SIZE, SFR, and SYMBOLIC parts replaces the function of the single Status Register soft module used in the keyboard interface of the previous chapter.



The **bit_timer** is built from an eight-bit up-counter with no carry input (**CUP8B**) coupled to a four-bit up-counter with a carry input (**CUP4BCI**). The carry output of **bit_timer_low** drives the carry input of **bit_timer_high**. The counters are clocked by the main clock of the CSoC by using the **BUSCLK** component. The counters are cleared whenever the **CLR** input goes high (i.e., whenever the **ps2_clock** signal is low).

At the end of a keyboard scan code transmission, the most-significant bit of the counter (**bit_timer11**) goes high and is logically ANDed with the high output of the **rcv_active** flip-flop. This clears the **rcv_active** flip-flop and sets the **rcv_int** flip-flop. The rising edge on the output of **rcv_int** clocks the **ps2_data_sreg** contents into the **rcvData** register. The **rcv_int** output also exits the keyboard interface through the **INTR** port. The value on the **INTR** port alerts the larger system when a keyboard scan code has been received.



#### Exporting the Netlist for a Schematic

After placing all the parts and adding the wiring in the **SCHEMATIC1** window, you can return to the **ps2port.opj** window and highlight the .\ps2port.dsn entry. This causes the schematic drawing tools to disappear from the right-edge of the Capture window and it enables the menu functions for exporting the schematic netlist to FastChip.



To begin exporting the keyboard interface netlist, click on the Tools⇒Create Netlist... menu item as shown below.



Click on the EDIF 200 tab in the **Create Netlist** window that appears. FastChip imports modules from EDIF netlists. The only option box that should be checked is the one which outputs pin names instead of numbers in the EDIF file. Then use the Browse button to select the orcad folder under the Chap41 project folder. The keyboard interface netlist will be stored in the PS2PORT.EDN EDIF file in this folder. Click on OK to begin the export process.

Create Netlist	×
EDIF 200 PSpice SPICE VHDL Verilog Layo	ut INF Other
Part Value Combined property string: {Value}	PCB Footprint Combined property string: {PCB Footprint}
Options ☐ Allow <u>n</u> onEDIF characters ☑ Output <u>p</u> in names (instead of pin numbers) ☐ <u>D</u> o not create "external" library declaration ☐ Output D <u>e</u> signator constructs	<ul> <li>Output net properties</li> <li>Output part properties</li> <li>Output pin properties</li> </ul>
Netlist <u>F</u> ile: □ ⊻iew Output C:\CSOC_EXAMPLES\CHAP41\ORCAD\PS2PORT	T.EDN <u>B</u> rowse
	OK Cancel Help

Once the netlist has been exported as an EDIF file, you can see it listed in the File tab of the **ps2port.opj** window under the Outputs folder.



#### Importing a Netlist into a FastChip Project

To begin importing the keyboard interface to your FastChip project, click on the

**Import EDIF** icon in the FastChip project window toolbar. **The Import EDIF module to the Imported Library** window appears.

Triscend I	FastChip: Import EDIF module to the Imported Library
	EDIF Design File ?
TD'30	File Name: < <u>EDIF Design File Name&gt;</u> Browse
	Automatically add Module Instance to "Configurable System Logic" area
۲	Module Name of Imported Module (optional) ?
	Automatically assign Module Name
	Module Name: <a href="https://www.www.www.www.www.www.www.www.www.w</td>
	✓ OK X Cancel ? Help

Click on the Browse button and steer your way into the orcad folder under the Chap41 project folder.

Triscend Fa	astChip File Browse	? ×
Look <u>i</u> n:	🔄 orcad 💽 🔁 📑 🗐	
	🛃 Desktop	
	🔜 My Computer	
	🛃 3½ Floppy (A:)	
	🗊 (C:)	
	CSoC_Examples	
	🗀 Chap41	
	📾 orcad	
	🗊 (D:)	
	🖃 Removable Disk (E:)	
1	▲ (F:)	
File <u>n</u> ame:	Se Network Neighborhood	I
Files of type	e: All Files (*.*)	31

You should highlight the PS2PORT.EDN EDIF file and click on the Open button.

Triscend FastC	Chip File Browse		? ×
Look <u>i</u> n: 🔂 o	prcad	- 🗈 🖄 😁	
PS2PORT.E			
	v		
File <u>n</u> ame:	PS2PORT.EDN		<u>O</u> pen
Files of <u>type</u> :	All Files (*.*)	•	Cancel

Now the path to the EDIF file should be listed in the File Name box and the Module Name field has been automatically updated with the name of your OrCAD project.

Triscend	FastChip: Import EDIF module to the Imported Library
****	EDIF Design File ? File Name: C:\CSoC_Examples\Chap41\orcad\PS2PORT.EDN & Browse Automatically add Module Instance to "Configurable System Logic" area Module Name of Imported Module (optional) ? Automatically assign Module Name Module Name: PS2PORT
	✓ OK X Cancel ? Help

Once you click on OK, the EDIF file import process begins. The progress is reported in the lower right-hand corner of the FastChip project window.

📈 Triscend FastChip: Cl	hap41 (Target Device: TE505S16-25L)
<u>M</u> File View Tools Help	
Project Library	DIF 1/0 Editor Generate Bind Download Debug Log Stop
Triscend Library ? X	Dedicated Resources ?
Archive Cisplays Cicogic Modules	Image: Clocks     Imag
Peripherals     CSI Bus	Configurable System Interconnect (CSI) Bus 💡
► I/O ► Memory	Configurable System Logic ? 🦭
Communications	
	Programmable I/O Pins ? Add I/O + 🏹
	D→ D→ D→ ps2_data ps2_clock LED
- 💎 Resource Estimate	<u>?</u>
❶gCSL Cells: 7/512 1	🕱 🖭 1/0 Pins: 9/60 📲 15 % 🕞 CSI Selectors: 1/32 📑 🕉 🚱 Performance: 25 MHz
Importing File Type [EDI	F]

You should receive the following status message after the import process completes. Click on OK to proceed.



Now you should see a new entry labeled Imported in the Triscend Library area. Expanding this entry shows that the keyboard interface circuit has been added to the library.

🚜 Triscend FastChip: Chap41 (Target Device: TE505S16-25L)				
📈 File View Tools Help				
Project Library Import EDIF I/O Editor Generate Bind Download Debug Log Stop				
Triscend Library ?X 2 Dedicated Resources ?				
<ul> <li>Archive</li> <li>Displays</li> <li>Logic Modules</li> </ul>	MCU Clocks MIU Timer_0 Timer_1 Timer_2 UART Interrupts Watchdog			
▶ ☐ Peripherals ▶ ☐ CSI Bus	Configurable System Interconnect (CSI) Bus ?			
▶ ☐ I/O ▶ ☐ Memory	Configurable System Logic 🙎 🥡			
Communications				
- SPACE CONT	* * *			
Programmable I/O Pins ? Add I/O + 🏹				
	ps2_data ps2_clock LED			
Resource Estimate ?				
UCSL Cells: 7/512 1 2 1 2 1 2 1 2 1 5 2 0 1 5 2 0 CSI Selectors: 1/32 3 2 1 2 1 2 1 2 1 2 1 2 2 5 MHz				
✔Ready.				

#### Using the Imported Module

You can drag your newly-imported **PS2PORT** module into the Configurable System Logic area just like any other soft module.

🚜 Triscend FastChip: Chap41 (Target Device: TE505S16-25L)				
📈 File View Tools Help				
Project Library Import EDIF I/O Editor Generate Bind Download Debug Log Stop				
Triscend Library ?X Dedicated Resources ?				
Archive	MCU Clocks MIU Fimer_0 Fimer_1 UART Interrupts Watchdog			
▶				
<ul> <li>CSI Bus</li> <li>I/O</li> <li>Memory</li> <li>Communications</li> <li>Imported</li> </ul>	Configurable System Interconnect (CSI) Bus 👔			
	🖉 🐟 Configurable System Logic 🙎 🤯 —			
	IEDPort PS2PORT_A			
- B≫PS2PORT	An imported module.			
	Programmable I/O Pins ? AddI/O - 🥡			
	ps2_data ps2_clock LED			
Resource Estimate ?				
Un CSL Cells: 39/512 8 2 № 1/0 Pins: 9/60 15 2 CSI Selectors: 2/32 6 2 100 Performance: 25 MHz				
✓Ready.				

Then click on the **PS2PORT_A** module to make the **Module - PS2PORT** window appear. The input and output ports in the original schematic are arranged along the left and right side of the module block, respectively. The port names in the schematic are also used as the default labels for the input and output nets.

Triscend FastChip: Module - PS2PORT (PS2PORT_A)					
	Component Name: PS2PORT_A				
	<pre> </pre> PS2_DATA INTR    INTR				
	PS2_CLOCK				
Apply VOK X Cancel ? Help					

You can rename the module to **PS2PORT**. The **PS2_DATA** input of the keyboard interface should be connected to the **ps2_data** input pin of the FastChip project. The keyboard clock signal arrives through the **ps2_clock** input pin and this drives the **GBuf0** global clock buffer. The output of **GBuf0** drives the clock input of the keyboard interface module. The output which indicates the presence of a keyboard scan code is connected to interrupt signal 0 (**INTR0**) of the 8032 MCU.

Note that some of the keyboard interface module I/O is not explicitly visible. The **BusClock** is implicitly input to the keyboard module by the inclusion of the **BUSCLK** component in the schematic. The interface to the CSI address and data buses is also implicitly defined by the use of the **DR7_OE**, **SELECTOR**, **SIZE**, **SFR**, and **SYMBOLIC** components.



After clicking OK in the **Module - PS2PORT** window, your FastChip project window should appear as follows.

📈 Triscend FastChip: Chap41 (Target Device: TE505S16-25L)				
📈 File View Tools Help				
Project Library Import EDIF I/O Editor Generate Bind Download Debug Log Stop				
Triscend Library ? X	Triscend Library ?X Dedicated Resources ?			
<ul> <li>▲ Archive</li> <li>▲ Displays</li> <li>▲ Logic Modules</li> </ul>	Archive Displays IL caic Modules			
▶  Peripherals	Configurable System Interconnect (CSI) Bus			
CSI Bus	Configurable system interconnect (CSI) Bus			
Memory	📸 🐟 Configurable System Logic 🙎 🤯 ———————————————————————————————————			
Communications	ledPort PS2PORT	Press and and		
- St PSZPORI	\$			
	Programmable I/O Pins ? AddI/O + 🦭			
	ps2_data ps2_clock LED			
TResource Estimate ?				
10 CSL Cells: 39/512 8 2 10 Pins: 9/60 15 2 CSI Selectors: 2/32 6 2 10 Performance: 25 MHz				
✔Ready.				

The modules and their interconnections have been instantiated. The pin assignments should already be set as shown in Table 13.

Signal	Pin	CSoC Board Resource
ps2_clock	47	PS/2 clock input
ps2 data	51	PS/2 data input
. <u> </u>	35	LED digit segment A
	39	LED digit segment B
	43	LED digit segment C
	11	LED digit segment D
	41	
	40	LED digit segment E
	34	LED digit segment F
LED.6	36	LED digit segment G

Table 13: Pin assignments and functions for the keyboard interface design.

Press the Generate icon on the toolbar and FastChip will create the chap41.h header file (Listing 9). Now you can check how the FastChip software handled the **SYMBOLIC** address component attached to the **SELECTOR**. Note that on line 34 the register which holds the scan code is declared to be of type sfr. So it was placed in the special function register address space of the 8032 as you specified by using the **SFR** component in your schematic. The name of the register was formed by concatenating the module name (PS2PORT) with the alias for the net that connects the **SYMBOLIC** component to the **SELECTOR** component (RCVDATA).

## Listing 9: Top of the header file generated by the FastChip software for the keyboard interface.

```
// Generated 5/28/00 9:12 AM By FastChip Version 1999 Build 30
11
11

// ----- GENERATED CODE ------
// ------
// The code in this header file was generated automatically for your
// project by Triscend FastChip. Please DO NOT EDIT this header file.
// It will be overwritten the next time FastChip generates code for
// your project.
11
//===== Required symbol and macro definitions =======
#ifdef PROTOTYPE ONLY
define CHAR XDATA(name, location) extern volatile unsigned char xdata name;
define CHAR ARRAY XDATA(name,location,size) extern volatile unsigned char
xdata name[size];
#else
define CHAR XDATA(name,location) volatile unsigned char xdata name at
location;
define CHAR_ARRAY_XDATA(name,location,size) volatile unsigned char xdata
name[size] at location;
#endif
//====== BEGIN SOFT MODULE REGISTER DECLARATIONS ======
//----- Module ledPort
 CHAR XDATA (ledPort,0xefff)
//----- Module PS2PORT
 sfr PS2PORT RCVDATA = 0x9a;
//====== END SOFT MODULE REGISTER DECLARATIONS ======
```

The next step is to write your application code. Create a keil folder within your Chap41 FastChip project folder. Then start the Keil IDE and add the C code in Listing 10 to the *chap41* Keil project. The main routine (lines 3-7) just initializes the 8032 MCU and enters an infinite-loop. All the work is actually done in the displayPs2Data interrupt subroutine (lines 28-47). This subroutine is called when the keyboard interface

©2000 by XESS Corp.

generates an INT0 interrupt (interrupt identifier 0). The interrupt subroutine reads the keyboard scan code from the **PS2PORT_RCVDATA** register on line 33 and this also clears the interrupt flag in the keyboard interface. Then the scan codes in the table defined on lines 13–26 are searched. If a matching scan code is found in the table, the subroutine writes the associated LED segment activation pattern to the **IedPort** register. This displays the digit for the key that was pressed. (The table only has the scan codes for the digits 0–9.) If the received scan code can't be found in the table, the subroutine displays an E on the LED digit.

```
Listing 10: Keyboard interface interrupt-handling code.
```

```
#include "..\Chap41.h"
1
2
3
 main()
4
 {
5
 Chap41 INIT();
6
 while(1);
7
 }
8
9
10
 #define ERROR 0x79;
11
12
 // translate keyboard scan codes to LED segment activations
13
 typedef struct{ unsigned char ps2Data, led; } ps2XlateEntry;
14
 ps2XlateEntry ps2XlateTbl[] =
15
 {
16
 { 0x16, 0x06 }, //
 "1"
17
 { 0x1E, 0x5B }, //
 "2"
 { 0x26, 0x4F }, //
18
 "3"
19
 { 0x25, 0x66 }, //
 "4"
 0x2E, 0x6D }, //
20
 "5"
21
 0x36, 0x7D }, //
 "6"
22
 { 0x3D, 0x07 }, //
 "7"
23
 { 0x3E, 0x7F }, //
 "8"
24
 { 0x46, 0x6F }, //
 "9"
25
 { 0x45, 0x3F } // "0"
26
 };
27
28
 static void displayPs2Data() interrupt 0 using 0
29
 {
30
 unsigned int i;
31
 unsigned char c;
32
33
 c = PS2PORT RCVDATA; // get the scan code
34
 // (also clears interrupt)
35
36
 // search the translation table for the scan code
37
 for(i=0; i<sizeof(ps2XlateTbl)/sizeof(ps2XlateEntry); i++)</pre>
```

```
38
 if(ps2XlateTbl[i].ps2Data == c)
39
 { // found a matching scan code in the table so
40
 // display the digit on the LED
41
 ledPort = ps2XlateTbl[i].led;
42
 return;
43
 }
44
45
 // no matching scan code was found, so display "E"
46
 ledPort = ERROR;
47
 }
```

Once you set the compiler and linker options as you did in the previous chapters, you can compile and link the *Chap41* Keil project. Then re-enter the FastChip project window and bind your design. Download the keyboard interface circuitry and the 8032 program in the Chap41.HEX file to your CSoC Board.



Finally, use dScope to establish a debugging link to the CSoC Board and then reset and execute the application program. At this point, you should be able to type on the numeric keys of a keyboard attached to the PS/2 port of your CSoC Board and see the numbers appear on the LED digit.