

Testing REGISTERS AND PERIPHERALS

Objectives

At the end of this section, you will be able to

• Use register contents as part of expected pass/fail testing outcomes





Testing

REGISTERS AND PERIPHERALS

1	Testing the MCU's peripherals	3-4
2	Testing peripherals	5-7
	Navigate to register Watch register's memory Test cases	5 6 7
3	Summarv	8-9

Summary



In this unit we will examine how to use the contents of the microcontroller's Special Function Registers (SFRs) as the *Expected* value of a testIDEA test case.

Such tests could be considered to digress towards Integration Testing rather than pure Unit Testing. However, in the world of embedded development and working close to the hardware, such testing is required, often when testing the functionality of peripheral drivers.

Here we will test that the function *digitalWrite()* works correctly, the function we have been using to turn a specified digital GPIO pin on or off.

void loop() {

digitalWrite(13, HIGH); delay(500); digitalWrite(13, LOW); delay(500); One potential method would be to create a hardware-in-the-loop (HIL) platform and check that the associated GPIO pin actually acquires the expected value. This would, however, require the GPIO to be connected to an IOM input of the BlueBox[™].

Instead we will execute the function and then check that the register used to control the GPIO pin has been changed appropriately.



l

In the winIDEA workspace, one way to examine the status of the GPIO pin that controls the LED is using the Special Function Registers (SFR) window.

The LED is attached to a pin under the control of register PIOB_ODSR, as shown in the screenshot opposite.

From the SFR window it is possible to determine the address of this register in memory, namely 0x400E1038.

	SH Test P	lugins	Tools	Winde	ow He	lp											
Project Workspace		▼ □ ↔	n x		exercis	e_06.cp	b										4
x Fitter					7 / 8 ♥ 9 10 11 } 12 13 / ↓ 15 16 17 18 19 20	/ the oid se // pir / the oid lo dig del dig del	setup initia Mode(1 loop f pop() { yitalWr Lay(500 yitalWr Lay(500	<pre>functio { lize c 3, OUT functio itte(13); itte(13); </pre>	on run ligital PDT); n runs , HIGH // , LOW) //	<pre>s once when pin 13 as over and o); // tur wait for a ; // tur wait for a</pre>	you press reset or power an output. ver again forever n the LED on (HIGH is the second n the LED off by making t second	the board voltage 1	l .evel) : LOW				
Symbols			_		21												
			1 Y 1	Variable	<							Watch					~
n		• • .		Context	5 t							Real Providence	0xF 🖦 🕾 🛄			~	
				Name		Va	lue			Type	Address Error	Name		Value	Typ	e	Address
																-	
												:0x400	E1038,u		ur	nsigned long	(Virtual) 400
												:0x400 @PIOB_	E1038,u ODSR		ur ur	nsigned long nsigned long	(Virtual)400 400E1038
			>							ß		:0x400 @PIOB_	E1038,u ODSR		ur	nsigned long nsigned long	(Virtual)4001 400E1038
d Find In Files Tools Script P	rogress		>	🧳 Lo	ocals	🧳 this				Ģ		: 0x400 @PIOB	E1038, u ODSR 6J Watch2	6J Rt.Watch1	ur سالت ها Rt.Watch2	nsigned long	(Virtual)4001 400E1038
Id Find In Files Tools Script P	rogress		>	🧳 Lo	ocals	🧳 this				Ç.		:0x400 @PIOB_ &J Watch	E1038, u ODSR 6J Watch2 Disassembly	୶ Rt.Watch1	یر ها Rt.Watch2	nsigned long nsigned long	(Virtual)4001 400E1038
Id Find In Files Tools Script P Value	rogress Values	;	>	🧳 Lo	ocals	🧳 this				Address	Description	:0x400 @PIOB_ & Watch	E1038, u ODSR 64 Watch2 Disassembly 22 42 42 42	6 J Rt.Watch1 ⇒ ↑ ↓ [wr wr 64 Rt.Watch2	nsigned long nsigned long	(Virtual)4001 400E1038
Id Find In Files Tools Script P Value	rogress Values		>	🥔 La	ocals	🧭 this				Address	Description	:0x400 @PIOB	E1038, u ODSR GJ Watch2 Disassembly G + + + + + + + + + + + + + + + + + + +	&d Rt.Watch1 ⇔ ✿ ₽ [ata Disa	GJ Rt.Watch2	ssigned long ssigned long	(Virtual)4001 400E1038
ld Find In Files Tools Script P Value 10B	rogress Values P0	P1 F	22	₽3	ocals P4	¢ this	P6 P	7 P8	P9	Address 400E1034	Description Clear Output Data Register	:0x400 @PIOB	E1038, u ODSR Uisassembly Address Dr	GJ Rt.Watch1 → ↑ ↓ [ata Disa	GJ Rt.Watch2	signed long signed long	(Virtual) 4000 400E1038
Id Find In Files Tools Script P Value PIOB ● ◆ PIOB_CODR ● ◆ PIOB_ODSR	rogress Values P0 P0	P1 F	> 	₽3 P3	P4	 this P5 P5 	P6 P	7 P8	P9 P9	Address 400E1034 400E1038	Description Clear Output Data Register Output Data Status Register	:0x400 @PIOB_ & Watch * • • • • ×	E1038, u ODSR Disassembly Address Di	&J Rt.Watch1 ⇒ 1 ↓ [ata Disa	GJ Rt.Watch2	signed long	(Virtual) 4000 400E1038
Id Find In Files Tools Script P Value Value 10B	rogress Values P0 P0 P0	P1 F P1 F	> -2 -2 -2	 P3 P3 P3 	P4 P4 P4	 this P5 P5 P5 P5 	P6 P7 P6 P7 P6 P7	7 P8 7 P8 7 P8	P9 P9 P9	Address 400E1034 400E1038 400E103C	Description Clear Output Data Register Output Data Status Register Pin Data Status Register	:0x400 @PIOB	E1038, u ODSR Disassembly Address D	sJ Rt.Watch1 ↔ ↑ ↓ [ata Disa	sJ Rt.Watch2	signed long	(Virtual) 400 400El038
d Find In Files Tools Script P Value 108 ● ♥ PIOB_CODR ● ♥ PIOB_CDDR ● ♥ PIOB_PDSR ● ♥ PIOB_ER	P0 P0 P0 P0 P0 P0	P1 F P1 F P1 F	> 2 2 2 2 2 2 2 2	 L P3 P3 P3 P3 	P4 P4 P4 P4 P4	 this P5 P5 P5 P5 	P6 P7 P6 P7 P6 P7 P6 P7	7 P8 7 P8 7 P8 7 P8	P9 P9 P9 P9	Address 400E1034 400E1034 400E103C 400E103C	Description Clear Output Data Register Output Data Status Register Pin Data Status Register Interrupt Enable Register	:0x400 @PIOB	E1038, u ODSR 64 Watch2 Disassembly 24 ddress Dr	sJ Rt.Watch1 ⇒ I ♣ [ata Disa	s Registers	signed long	(Virtual)400 400E1038
d Find In Files Tools Script P Value IOB PIOB_CODR PIOB_CDDR PIOB_DDSR PIOB_PDSR PIOB_LER PIOB_LER PIOB_DR	rogress Values P0 P0 P0 P0 P0 P0 P0	P1 F P1 F P1 F P1 F	> 2 2 2 2 2 2 2 2 2 2 2	 P3 P3 P3 P3 P3 P3 P3 	P4 P4 P4 P4 P4 P4 P4	 this P5 P5 P5 P5 P5 P5 	P6 P7 P6 P7 P6 P7 P6 P7 P6 P7	7 P8 7 P8 7 P8 7 P8 7 P8 7 P8	P9 P9 P9 P9 P9 P9	Address Address 400E1034 400E1038 400E103C 400E1040 400E1044	Description Clear Output Data Register Output Data Status Register Pin Data Status Register Interrupt Enable Register Interrupt Disable Register	:0x400 @PIOB_ &J Watch	E1038, u ODSR	sJ Rt.Watch1 → 1 ♣ [ata Disa	6J Rt.Watch2	signed long	(Virtual)400 400El038
d Find In Files Tools Script P Value IOB	rogress Values P0 P0 P0 P0 P0 P0 P0 P0 P0	P1 F P1 F P1 F P1 F P1 F P1 F	>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>	 P3 P3 P3 P3 P3 P3 P3 P3 	P4 P4 P4 P4 P4 P4 P4 P4	 this P5 <li< td=""><td>P6 P P6 P P6 P P6 P P6 P P6 P</td><td>7 P8 7 P8 7 P8 7 P8 7 P8 7 P8 7 P8</td><td>P9 P9 P9 P9 P9 P9 P9</td><td>Address Address 400E1034 400E1038 400E103C 400E1040 400E1044 400E1048</td><td>Description Clear Output Data Register Output Data Status Register Pin Data Status Register Interrupt Enable Register Interrupt Disable Register Interrupt Mask Register</td><td>÷0x400 @PIOB GJ Watch</td><td>E1038, u ODSR Disassembly Address D</td><td>49 Rt.Watch1 → ↑ ↓ [ata Disa</td><td>s Registers</td><td>signed long</td><td>(Virtual)400 400E1038</td></li<>	P6 P P6 P P6 P P6 P P6 P P6 P	7 P8 7 P8 7 P8 7 P8 7 P8 7 P8 7 P8	P9 P9 P9 P9 P9 P9 P9	Address Address 400E1034 400E1038 400E103C 400E1040 400E1044 400E1048	Description Clear Output Data Register Output Data Status Register Pin Data Status Register Interrupt Enable Register Interrupt Disable Register Interrupt Mask Register	÷0x400 @PIOB GJ Watch	E1038, u ODSR Disassembly Address D	49 Rt.Watch1 → ↑ ↓ [ata Disa	s Registers	signed long	(Virtual)400 400E1038
d Find In Files Tools Script P Value 108 ■ ◆ PIOB_CODR ■ ◆ PIOB_ODSR ■ ◆ PIOB_PDSR ■ ◆ PIOB_IER ■ ◆ PIOB_IER ■ ◆ PIOB_IRR ■ ◆ PIOB_ISR	rogress Values PO PO PO PO PO PO PO PO	P1 F P1 F P1 F P1 F P1 F P1 F P1 F	>	 L P3 	P4 P4 P4 P4 P4 P4 P4 P4 P4 P4	 this P5 <li< td=""><td>P6 P7 P6 P7 P6 P7 P6 P7 P6 P7 P6 P7 P6 P7</td><td>7 P8 7 P8 7 P8 7 P8 7 P8 7 P8 7 P8 7 P8</td><td>P9 P9 P9 P9 P9 P9 P9 P9 P9</td><td>Address 400E1034 400E1038 400E103C 400E1040 400E1044 400E1048 400E104C</td><td>Description Clear Output Data Register Output Data Status Register Pin Data Status Register Interrupt Enable Register Interrupt Disable Register Interrupt Mask Register Interrupt Status Register</td><td>÷0x400 @PIOB_ ←J Watch</td><td>E1038, 11 ODSR Disassembly Address D</td><td>≪J Rt.Watch1</td><td>s Registers</td><td>signed long</td><td>(Virtual)400 400El038</td></li<>	P6 P7 P6 P7 P6 P7 P6 P7 P6 P7 P6 P7 P6 P7	7 P8 7 P8 7 P8 7 P8 7 P8 7 P8 7 P8 7 P8	P9 P9 P9 P9 P9 P9 P9 P9 P9	Address 400E1034 400E1038 400E103C 400E1040 400E1044 400E1048 400E104C	Description Clear Output Data Register Output Data Status Register Pin Data Status Register Interrupt Enable Register Interrupt Disable Register Interrupt Mask Register Interrupt Status Register	÷0x400 @PIOB_ ←J Watch	E1038, 11 ODSR Disassembly Address D	≪J Rt.Watch1	s Registers	signed long	(Virtual)400 400El038



The address of the register can be added to the Real-time Watch window. You will see the value updating as the code executes. The register can be accessed in two ways:

:0x400E1038,u – the address and data type (updates in "real time")

@PIOB_ODSR - the name of the register associated with the pin being controller (doesn't update in real time by default - see following slide)

We see that when the LED is on, PIOB ODSR == 0×08000000 , and when off it is 0×00000000 .

Vatch						▼ □ ↔	n x				
84 84 🔤					~						
Name			Value		Туре	Address					
:0x400E1	038,u		0x0800	0000	unsigned long	(Virtual) 400E1038	^				
@PIOB_ODSR			0x0800	0000	unsigned long	400E1038					
							~				
60 Watch1	6J Watch2	64 R	t.Watch1	6J Rt.Watch2							



Be careful when adding registers in the "Real-time Watch" window as sometimes the mere act of reading/writing the register can cause its contents to change, such as a register that clears its bits after having been read.



This default functionality regarding real time updates for SFRs can be changed if required. Simply open the Tools -> Options dialog and change the *Memory Access* option to *Real Time*.

	ay Jin w	indow	Callstack Wir	ndow	Terminal
Only visible registers All registers		Memory Ac Monito Real T	ccess r ime (if availal	ble)	
Display <u>N</u> umeric values <u>Description</u> Description <u>wi</u> th valu Display short names Display radix prefix	e	Note: Rea of SFRs is Some regi reading. A periodic can disrup behaviour	I-time access not recomm ster's state is refresh by th t the regular	s / upda ended. affected e debug applicat	te 1 by iger ion
Sub-register display None Values Names and values		Abort refres	h ter after if access end	5 1] s] s s



Testing the *digitalWrite()* function in testIDEA is quite straightforward. Simply create a test that calls the function with the required parameters (pin number 13 and '0' or '1' for the logic level desired).

In the "*Expected*" field we simply mask for the desired bit in the target register and compare the outcome with the expected outcome for the test to be considered to have passed.

This looks as shown opposite (testIDEA view is in *Table* mode).

۲		func							assert
	func		params	+					expressions
		0	* 1	+ ×					0
0	³¹ digitalWrite	¹ 13	ⁱ 1		i	i	i	<u>i</u> i	(@PIOB_ODSR & 0x08000000) == 0x08000000
1	³ digitalWrite	⁴ 13	ⁱ 0		i	i	i	<u>i</u> i	(@PIOB_ODSR & 0x08000000) == 0
									+





SUMMARY

testl	DEA



- testIDEA can also use the contents of registers, rather than variables, for its *Expected* values.
- Tests pass or fail dependent on the results found in the peripheral registers.
- Approach can be used to develop tests for microcontroller peripheral drivers.

