Fase I del test funcional

Como se vio al inicio del capítulo, esta fase está compuesta de una serie de pruebas que pretenden comprobar el funcionamiento general del sistema completo.

Esta fase se caracteriza porque los resultados de las pruebas ya han sido obtenidos a priori en co-simulaciones software-hardware. Esto facilita la validación del test ya que bastará con comparar los resultados obtenidos por simulación con los obtenidos por ejecución real.

La Fase I quedará dividida en dos sub-fases:

La sub-fase 1.1 que consiste en la ejecución del software interno del DTC. Bastará con dar alimentación al chip y ver los mensajes que se exteriorizan por el puerto paralelo del chip. El software a ejecutar se denomina DTC_A.

La sub-fase 1.2 consistente en la ejecución del software DTC_B, pretende validar el funcionamiento general del sistema desde la ejecución en memoria externa de dicho software.

Fase 1.1

Esquema hardware empleado

Los elementos hardware necesarios para llevar cabo la prueba serán:

- El chip de pruebas: DTC
- Un analizador lógico conectado al puerto paralelo del DTC.

Descripción funcional de cada rutina

La Rutina Reset_EH

Se encargará de realizar las siguientes tareas:

- Iniciará todos los registros internos del ARM
- Iniciará los punteros de pila de los modos empleados.
- Inhabilita el watchdog.
- Habilita la FIQ para permitir un mecanismo que saque el control a memoria externa. A partir de esto momento es posible pasar la ejecución a memoria externa provocando una interrupción FIQ desde el exterior.
- Chequea la RAM interna.
- Habilita el Watchdog
- Pasa el control a la rutina 1

La Rutina de Test 1

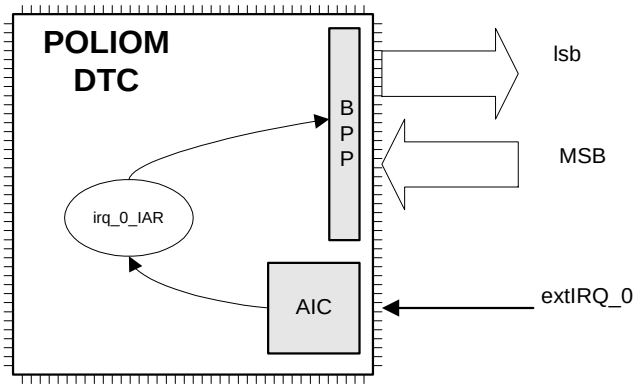
Realiza una comprobación simple del funcionamiento del Controlador de Memoria Interna (IMC).

La prueba consiste en transferir un buffer de ROM interna a RAM interna de 1 Kbyte leídos y escritos en word (32-bits).

La Rutina de Test 2

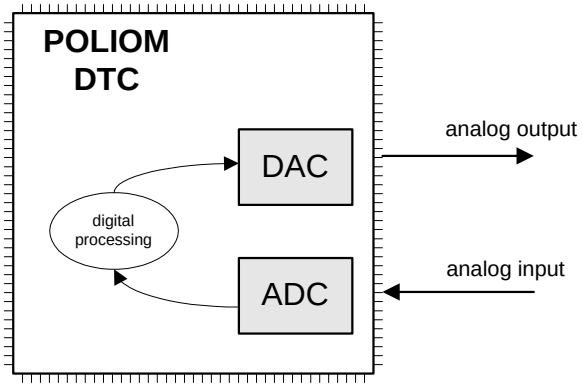
La prueba que se lleva a cabo en esta rutina tiene por objetivo poder chequear el mecanismo de interrupciones disponible en Policom. Para ello se programará el chip para que uno de sus pines de interrupción externa pueda provocar interrupción del procesador interno del DTC.

El observador externo podrá chequear el correcto funcionamiento de la rutina mediante los mensajes interactivos que esta proporciona: el observador podrá colocar un valor en los pines más significativos del puerto paralelo y, tras provocar interrupción por la IRQ externa, comprobará que el valor se copia en los pines menos significativos del puerto.



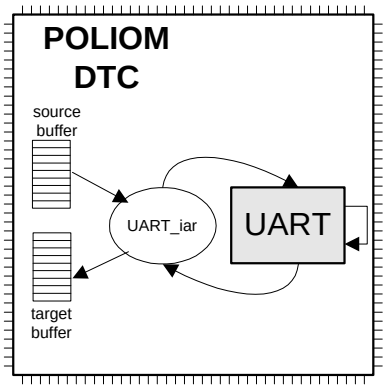
La Rutina de Test 3

Esta rutina chequea el funcionamiento de los convertidores del DTC. Empleando un temporizador, el chip irá muestreando una entrada analógica y procesando digitalmente esta señal. El resultado de este procesamiento se mostrará de forma analógica.



La Rutina de Test 4

Esta rutina chequea el funcionamiento del puerto serie asícrono (UART). Para ello se configurará la UART en modo “loop-back” y se enviará un buffer que al ser recibido se chequeará para comprobar el correcto funcionamiento.



La Rutina de Test 5

Rutina encaminada a chequear el mecanismo de atención de Interrupciones Software. La rutina realiza una llamada a la interrupción software número 2.La rutina de atención de interrupción software (SWI_IAR) desplazará un mensaje por el puerto paralelo.

La Rutina de Test 6

Comprueba el funcionamiento del SPI. La forma de hacerlo es transmitiendo y recibiendo un buffer de 32 bytes a una velocidad de 1Mbps. El resultado del chequeo se muestra por el puerto.

Control de versiones

A continuación se especifican las versiones de cada rutina involucrada en esta fase de test.

	Control de versiones	
	Versión	Fecha
Reset EH	3.0	Oct. 98
IRQ EH	3.0	Oct. 98
FIQ EH	1.0	Jun. 98
SWI EH	1.0	Oct. 98
Rutina 1	1.2	Jul. 98
Rutina 2	2.0	Oct. 98
Rutina 3	2.0	Oct. 98
Rutina 4	1.0	Oct. 98
Rutina 5	1.0	Oct. 98
Rutina 6	1.0	Oct. 98

Hipótesis de funcionamiento

Existen ciertas consideraciones a tener en cuenta sobre el código integrado en el DTC. Son las siguientes:

La ejecución se lleva a cabo en todo momento en modo ARM.

El modo de procesador será el SVC (supervisor), salvo en los instantes que se atiendan las interrupciones.

El Manejador de IRQ incluido en la ROM interna es la versión 2.0.

El mecanismo de interrupción se emplea de forma muy restrictiva. Las rutinas que emplean interrupción habilitan durante un tiempo la posibilidad de atender interrupción. Pasado ese tiempo se vuelve a inhabilitar las interrupciones en el propio procesador.

La interrupción del tipo FIQ se encuentra habilitada durante todo el proceso, permitiendo sacar la ejecución a memoria externa.

Los tiempos de ejecución, en especial los de espera, se encuentran calculados para la ejecución en el interior del DTC. Esto supone que

para poder ejecutar este software en memoria externa, será necesario ajustar ciertos parámetros relacionados con los tiempos de espera.

El SPI debe ser puenteado externamente para que la prueba pueda efectuarse.

Se sigue el mapa de memoria presentado en el capítulo 5.

Relación de celdas involucradas en cada test

La tabla que se presenta a continuación muestra las celdas del DTC involucradas en cada rutina interna de la ROM interna del DTC.

	DTC_A										
	Celdas ASB			Celdas APB							
	IMC	EMC	AIC	APB Bridge	UART	BPP	ADC	DAC	TIMER	WD	SPI
Reset EH	■		■	■		■				■	
Rutina 1	■			■		■				■	
Rutina 2	■		■	■		■					
Rutina 3	■		■	■		■	■	■	■		
Rutina 4	■		■	■	■	■					
Rutina 5	■			■		■					
Rutina 6	■		■	■		■					■

Observabilidad

Generalmente se emplea el puerto paralelo para mostrar mensajes sobre la evolución interna del software. Adicionalmente se pondrá emplear alguna de los pines de selección de periféricos externos.

Los diversos mensajes mostrados por el DTC son los siguientes:

	DTC_A					
	Puerto paralelo (BPP)				**Selección periférico externo**	
	tº	Mensaje	Significado	Etiqueta software	Nº	Significado
Reset EH	4,3 μs.	0xA5	Inicio de actividad en el chip.	MSG_RESET_1	-	-
	31.2 μs.	0xAA	Inicio de actividad del chip.	MSG_RESET_2	-	-
	58,1 μs.	0xAF	La RAM se está chequeando	MSG_RESET_TESTING_RAM		-
	1,67 ms.	0xAB	Test de RAM fallado.	MSG_RESET_RAM_TEST_FAILED	-	-
	1,67 ms.	0xA1	Test de RAM correcto	MSG_RESET_RAM_TEST_OK	-	-
Rutina 1	1,68 ms	0xB0	Inicio del test 1	MSG_TEST_ROUTINE_1	-	-
Rutina 2	1,73 ms	0xB1	Inicio del test 2	MSG_TEST_ROUTINE_2	1	Se activa cada vez que se entre en la IAR del EP1
Rutina 3	1,93 ms	0xB2	Inicio del test 3	MSG_TEST_ROUTINE_3	1	Al entrar en la IAR del timer.
Rutina 4	2,927 ms	0xB3	Inicio del test 4	MSG_TEST_ROUTINE_4	-	-
	6,95 ms	0x1B	Test de la UART correcto	MSG_UART_TEST_OK	-	-
	6,95 ms	0x1C	Test de la UART incorrecto.	MSG_UART_TEST_FAILED	-	-
Rutina 5		0xB4	Inicio del test 5	MSG_TEST_ROUTINE_5		
		0xCA	Mensaje que se irá sdesplazando por el puerto	MSG_SWI_TEST_5	-	-
Rutina 6		0xB5	Inicio del test 6	MSG_TEST_ROUTINE_6	-	-
	7,37 ms.	0x2B	Test del SPI correcto	MSG_SPI_TEST_OK	-	-
	7,37 ms	0x2C	Test del SPI incorrecto.	MSG_SPI_TEST_FAILED	-	-

Recursos de bajo nivel empleados

La siguiente tabla muestra los recursos software de bajo nivel asociados a la ejecución de cada rutina de test, entendiendo por tales aquellos que están relacionados con el tratamiento de excepciones.

	EH's				
	IRQ		SWI	FIQ	RESET(*)
	IID	Dispositivo	SWIID		
Reset EH	-	-	-	Pasa a habilitada	En ejecución
Rutina 1	-	-	-	Habilitada	-
Rutina 2	0xE	Externa 1	-	Habilitada	-
Rutina 3	0x5	Timer 2	-	Habilitada	-
Rutina 4	0x8	UART (tx & rx)	-	Habilitada	-
Rutina 5	-	-	0x2	Habilitada	-
Rutina 6	0xD	SPI (tx & rx)		Habilitada	-

(*) El Manejador de Excepción del tipo Reset se invocará en caso de provocarse otro tipo de excepción que no esté implementada en Policom.

Fase 1.2

Esquema hardware empleado

Los elementos hardware necesarios para llevar cabo la prueba serán:

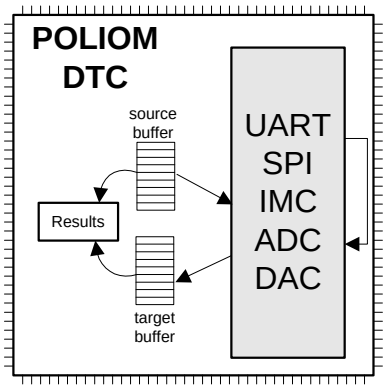
- El chip de pruebas: DTC
- Un analizador lógico conectado al puerto paralelo del DTC.
- La tarjeta de test DTC Test Board

Este software se cargará en ROM externa, y su ejecución se llevará a cabo por el DTC configurado para que arranque desde memoria externa.

Descripción funcional de cada rutina

Las rutinas que forman el software DTC_B tienen aproximadamente la misma estructura. Todas ellas toman un buffer de origen, lo procesan según la funcionalidad de cada dispositivo, y el resultado se almacena en un buffer destino. Las direcciones base de los buffers así como el tamaño de los mismos son programables.

La figura muestra el esquema que se sigue en general.



La Rutina UART_Test

Realiza un chequeo de la UART transfiriendo un buffer de tamaño variable. La configuración inicial de la UART emplea el modo “loop back”, si bien es posible realizar el chequeo en bucle abierto con un PC cerrando la recepción.

El cambio de configuración puede ser fácilmente alterable en el código de la rutina.

Esta rutina funciona acompañada de una IAR que gestionará los eventos que ocurran en la UART.

La Rutina SPI_Test

En este caso se realiza un chequeo de la celda de gestión de las comunicaciones SPI del DTC. AL igual que con el test anterio, el tamaño del buffer en un parámetro.

Inicialmente se realizará la prueba cortocircuitando la salida y la entrada del SPI.

La Rutina IMC_Test

Esta rutina chequea el funcionamiento del controlador de memoria interna, externa o ambos, dependiendo de configuración que se introduzca.

El test consistirá en transferir buffer de tamaño programable y cuyas direcciones de origen y destino son también parámetros que puede cambiar el programador.

La transferencia se lleva a cabo de tres formas dirferentes: byte a byte, halfword a halfword, y de word en word. De esta manera se estará chequeando el funcionamiento correcto de todos los modos de transferencia del sistema DTC.

Al final de la transferencia se chequearán los resultados.

La Rutina ADC_DAC_Test

Realiza un chequeo del funcionamiento de las celdas de conversión AD y DA.

El test consiste en extraer un valor analógico por el DAC, y a continuación lanzar una conversión ADC. Si se puentean externamente los pines correspondientes a los convertidores de forma adecuada, se estará realizando un doble conversión digital → analógica → digital. Los valores digitales extremos se compararán para comprobar el correcto funcionamiento de los dispositivos involucrados.

Control de versiones

A continuación se especifican las versiones de cada rutina involucrada en esta fase de test.

	Control de versiones	
	Versión	Fecha
Reset EH	3.0	Dic. 98
IRQ EH	4.0	Nov. 98
FIQ EH	-	-
SWI EH	-	-
UART TEST	1.2	Jul. 98
SPI TEST	2.0	Oct. 98
IMC TEST	2.0	Oct. 98
DAC_ADC TEST	1.0	Oct. 98

Hipótesis de funcionamiento

Existen ciertas consideraciones a tener en cuenta sobre el código de test DTC_B. Son las siguientes:

En simulación se ha empleado código ARM, lo cual hace parecer lógico que las primeras simulaciones se realicen en ARM. A continuación se generará código THUMB para ver cómo se comporta el sistema.

El modo de procesador será el SVC (supervisor), salvo en los instantes que se atiendan las interrupciones, que cambiará a IRQ.

El Manejador de IRQ incluido en el DTC_B es la versión 4.0.

Será necesario ajustar algunos parámetros relacionados con los tiempos máximos de espera para que funcionen correctamente las pruebas.

El SPI debe ser puenteado externamente para que la prueba pueda efectuarse.

El dac y el adc también deberán ser puenteados.

Relación de celdas chequeadas en cada test

La tabla que se presenta a continuación muestra las celdas del DTC a las que cada rutina intenta testar.

	DTC_B									
	Celdas ASB			Celdas APB						
	IMC	EMC	AIC	UART	BPP	ADC	DAC	TIMER	WD	SPI
Reset EH									■	
UART TEST			■	■						
SPI TEST			■							■
IMC TEST	■	■								
ADC_DAC TEST			■			■	■	■		

Observabilidad

El software DTC_0B utiliza dos formas diferentes de comunicarse con el exterior para mostrar sus mensajes de estado:

- El puerto paralelo. Se utiliza cada vez que se entra en una rutina de test
- La memoria RAM (interna o externa, según se configure). En la RAM se guardará la siguiente información: los flags que indicarán el éxito o fracaso de cada test, así como los datos que han sido transferidos por cada rutina.

Puede alterarse las etiquetas para que el resultado de los test sea mostrado por el puerto en vez de guardarse en RAM.

Los diversos mensajes mostrados por el DTC_B son los siguientes:

	DTC_B				
	RAM			BPP	
	Dirección de flag	Flag	Buffer origen/destino	Mensaje	Etiqueta
UART TEST	UART_TEST_FLAG_Addr	0xFF00 →OK 0x00FF →FAILED	UART_TEST_SOURCE_AREA_Begin UART_TEST_TARGET_AREA_Begin	0x2	MSG_UART_TEST
SPI TEST	SPI_TEST_FLAG_Addr	0xFF00 →OK 0x00FF →FAILED	SPI_TEST_SOURCE_AREA_Begin SPI_TEST_TARGET_AREA_Begin	0x4	MSG_SPI_TEST
ADC DAC TEST	DAC_ADC_TEST_FLAG_Addr	0xFF00 →OK 0x00FF →FAILED	DAC_ADC_TEST_SOURCE_AREA_Begin DAC_ADC_TEST_TARGET_AREA_Begin	0x8	MSG_DAC_ADC_TEST
IMC TEST	IMC_TEST_FLAG_Addr	0xFF00 →OK 0x00FF →FAILED	IMC_TEST_SOURCE_AREA_Begin IMC_TEST_TARGET_AREA_Begin	0x10	MSG_IMC_TEST

Recursos de bajo nivel empleados

La siguiente tabla muestra los recursos software de bajo nivel asociados a la ejecución de cada rutina de test, entendiendo por tales aquellos que están relacionados con el tratamiento de excepciones.

	EH's				
	IRQ		SWI	FIQ	RESET(*)
	IID	Dispositivo	SWIID		
Reset EH	-	-	-	Inhabilitada	En ejecución
UART Test	-	UART rx & tx	-	Inhabilitada	-
SPI Test	0xD	SPI rx & tx	-	Inhabilitada	-
ADC/DAC Test	0xD	Timer 1	-	Inhabilitada	-

IMC test	-		-	Inhabilitada	-

(*) El Manejador de Excepción del tipo Reset se invocará en caso de provocarse otro tipo de excepción que no esté implementada en Policom.

Parámetros configurables

El software de test DTC_B contiene en su cabecera una serie de parámetros que permiten ajustar el software a las necesidades de cada test.

Los parámetros más significativos son los siguientes:

- Mensajes del puerto paralelo → Se trata de los mensajes que las rutinas escriben en el puerto para indicar que se están ejecutando.
- Flags de test → Son los mensajes que se escriben para indicar si el test se ha completado con éxito o no.
- Tamaños de los buffers empleados → El tamaño de cada buffer empleado en cada prueba puede ser especificado por el programador.
- Direcciones de destino y origen de los buffers de prueba → Las direcciónes de los buffers antes mencionados pueden ser configurados tanto los de origen como los de destino.
- Zona de flag → Se trata de la zona de memoria donde se escriben el resultado de cada prueba, esto es, si ha tenido éxito o no. Inicialmente las direcciones elegidas se encuentran apuntando a la RAM, pero en cualquier caso pueden ser alteradas para que lo hagan al puerto paralelo.

La siguiente figura muestra la parte del código a la que hace referencia este apartado.

```
MSG_TESTING_RAM         EQU     0x1
MSG_UART_TEST           EQU     0x2
MSG_SPI_TEST            EQU     0x4
MSG_DAC_ADC_TEST        EQU     0x8
MSG_IMC_TEST            EQU     0x10
RAM_TEST_OK             EQU     0xFF00
RAM_TEST_FAILED         EQU     0x00FF
UART_TEST_OK            EQU     0xFF00
UART_TEST_FAILED        EQU     0x00FF
SPI_TEST_OK             EQU     0xFF00
SPI_TEST_FAILED         EQU     0x00FF
IMC_TEST_OK             EQU     0xFF00
IMC_TEST_FAILED         EQU     0x00FF
DAC_ADC_TEST_OK         EQU     0xFF00
DAC_ADC_TEST_FAILED     EQU     0x00FF
FLAG_AREA_Size          EQU     0x40
UART_TEST_Size          EQU     0x4     ;(64)
SPI_TEST_Size           EQU     0x80  ;(128bytes)
IMC_TEST_BYTES_Size     EQU     0x100   ;(256)
IMC_TEST_HALFWS_Size    EQU     0x80
IMC_TEST_WORDS_Size     EQU     0x40
IMC_TEST_Size           EQU
  (IMC_TEST_WORDS_Size*4)+(IMC_TEST_HALFWS_Size*2)+IMC_TEST_BYTES_Size
DAC_ADC_TEST_Size               EQU     0x8
EMC_TEST_Size           EQU     0x0
FLAG_TEST_Size          EQU     0x14
FLAG_TEST_AREA_Begin    EQU     INT_RAM_USER_DATA_Base
IMC_TEST_TARGET_AREA_Begin              EQU     FLAG_TEST_AREA_Begin+FLAG_AREA_Size
IMC_TEST_SOURCE_AREA_Begin              EQU     0x84
EMC_TEST_TARGET_AREA_Begin              EQU
  IMC_TEST_TARGET_AREA_Begin+IMC_TEST_Size
EMCI_TEST_SOURCE_AREA_Begin             EQU     0x84
DAC_ADC_TEST_TARGET_AREA_Begin          EQU
EMC_TEST_TARGET_AREA_Begin+EMC_TEST_Size
DAC_ADC_TEST_SOURCE_AREA_Begin          EQU     0x84
UART_TEST_TARGET_AREA_Begin             EQU
DAC_ADC_TEST_TARGET_AREA_Begin+DAC_ADC_TEST_Size
UART_TEST_SOURCE_AREA_Begin             EQU     0x84
SPI_TEST_TARGET_AREA_Begin              EQU
UART_TEST_TARGET_AREA_Begin+UART_TEST_Size
SPI_TEST_SOURCE_AREA_Begin              EQU     0x84
RAM_TEST_FLAG_Addr      EQU     FLAG_TEST_AREA_Begin
IMC_TEST_FLAG_Addr      EQU     FLAG_TEST_AREA_Begin+0x04
EMC_TEST_FLAG_Addr      EQU     FLAG_TEST_AREA_Begin+0x08
DAC_ADC_TEST_FLAG_Addr          EQU     FLAG_TEST_AREA_Begin+0x0C
UART_TEST_FLAG_Addr     EQU     FLAG_TEST_AREA_Begin+0x10
SPI_TEST_FLAG_Addr              EQU     FLAG_TEST_AREA_Begin+0x14
```