Application Note

DA16200 Pin Multiplexing

AN-WI-010

Abstract

This document provides information about the pin multiplexing in DA16200.

Contents

Figures

Tables

1 Introduction

DA16200 provides various interfaces to support many kinds of applications. It is possible to control each pin according to the required application in reference to the pin multiplexing illustrated in Figure 1. Pin control can be realized through register setting. This device can use a maximum of 16 GPIO pins (GPIOA[11:0], GPIOC[8:6], and GPIOA[15]) and each of the GPIO pins have various functions like I2C, SPI, etc. To determine which function should be selected in each GPIO pins, use the pin multiplexing. The four pins from GPIOA0 to GPIOA3 support analog input for ADC function in addition to digital signals, which also can be realized through register setting.

Figure 1 shows the functions available for each GPIOs. For example, if you want the I2C master function in DA16200, you can check which GPIOs support the I2C master function in this figure. You can see GPIOA[1:0], GPIOA[5:4], and GPIOA[9:8] support I2C master function.

DA16200 Pin Multiplexing

Pin	JTAG	Analog	SPI master	SPI slave	I2C master	I2C slave	SDIO slave	SDeMMC	BT coex	I2S	I2S_Clock	UART1	UART2	Muxed w/Analog	Pin State (nRESET=0)	Driving Strength (Default : 8mA)
GPIOA0		CH0		SPI_MISO	I2C_SDA	I2C_SDA				BCLK		TXD		Yes	I-PD	2/4/8/12mA
GPIOA1		CH1		SPI_MOSI	I2C_CLK	I2C_CLK		WRP		MCLK		RXD		Yes	I-PD	2/4/8/12mA
GPIOA2		CH2		SPI_CSB		I2C_SDA				SDO		TXD		Yes	I-PD	2/4/8/12mA
GPIOA3		CH3		SPI_CLK		I2C_CLK				LRCK	CLK_IN	RXD		Yes	I-PD	2/4/8/12mA
GPIOA4					I2C_SDA	I2C_SDA	CMD	CMD		BCLK		TXD/RTS		No	I-PD	2/4/8/12mA
GPIOA5					I2C_CLK	I2C_CLK	CLK	CLK		MCLK		RXD/CTS		No	I-PD	2/4/8/12mA
GPIOA6			SPI_CSB	SPI_CSB		I2C_SDA	D3	D3		SDO		TXD		No	I-PD	2/4/8/12mA
GPIOA7			SPI_CLK	SPI_CLK		I2C_CLK	D2	D2		LRCK		RXD		No	I-PD	2/4/8/12mA
GPIOA8			SPI_DIO0	SPI_MISO	I2C_SDA		D1	D1	BT_SIG0	BCLK				No	I-PD	2/4/8/12mA
GPIOA9			SPI_DIO1	SPI_MOSI	I2C_CLK		D0	D0	BT_SIG1	MCLK				No	I-PD	2/4/8/12mA
GPIOA10			SPI_DIO2	SPI_MISO				WRP	BT_SIG2		CLK_IN		TXD	No	I-PD	2/4/8/12mA
GPIOA11			SPI_DIO3	SPI_MOSI									RXD	No	I-PD	2/4/8/12mA
TCLK/GPIOA15	TCLK													No	I-PD	2/4/8/12mA
TMS	TMS													No	I-PU	2/4/8/12mA
UART_TXD														No	O	2/4/8/12mA
UART_RXD														No	I	2/4/8/12mA
GPIOC8	TDI													No	I-PD	2/4/8/12mA
GPIOC7	TDO												RXD	No	I-PD	2/4/8/12mA
GPIOC6	NTRST												TXD	No	I-PD	2/4/8/12mA

Figure 1: DA16200 Pin Multiplexing

2 PIN MUX Details

Figure 2 and Figure 3 show the PIN MUX layout. The color-coding information for the figures is the following:

- Black => input
- Red => output
- Violet => in/out

Figure 2 shows the functions available for the GPIOA group. The GPIOA[11:0] pin is available for pin multiplexing in the GPIOA group. (TCLK, pin#7, can also be used as a GPIOA[15])

For example, if you want to use the I2C master function in DA16200, you need a I2c clock and data signals. As described before, Figure 1 shows possible candidates for the I2C master function (GPIOA[1:0], GPIOA[5:4], and GPIOA[9:8]) and you can select one of them for your application.

Let's assume you select GPIOA[5:4] as a I2C master function. GPIOA[4] and GPIOA[5] are located in the third row from the bottom in Figure 2 and there is a I2C master at the same row. This means that GPIOA[4] and GPIOA[5] are defined as the I2C master signals. In this case, I2C_SDA and I2C_CLK respectively. Check the value for the I2C master; for our example it is '5'. This value should be set to the register FSEL_GPIO1[11:8]. See Section 3 for information on the register map.

Note that, both GPIOA[4] and GPIOA[5] are configured as the I2C signals with one single register value of '5'. Two pins are defined by one single register value.

The _fc9k_io_pinmux (UINT32 mux, UINT32 config) API in the SDK is for this pin mux configuration. After this API is called, check if the corresponding register is set to the value defined in Figure 2.

The "mux" parameter in the API is defined in <da16200_ioconfig.h> as below:

```
#define       PIN_AMUX      0
#define       PIN_BMUX      1
#define       PIN_CMUX      2
#define       PIN_DMUX      3
#define       PIN_EMUX      4
#define       PIN_FMUX      5
#define       PIN_IMUX      6
#define       PIN_JMUX      7
#define       PIN_KMUX      8
#define       PIN_HMUX      9
#define       PIN_LMUX      10
#define       PIN_MMUX      11
#define       PIN_NMUX      12
#define       PIN_PMUX      13
#define       PIN_QMUX      14
#define       PIN_RMUX      15
#define       PIN_SMUX      16
#define       PIN_TMUX      17
#define       PIN_UMUX      18
#define       PIN_ALLMUX    19
```

AMUX is in the bottom row of Figure 2 and includes GPIOA[0] and GPIOA[1].

BMUX is in the second row from the bottom in Figure 2 and includes GPIOA[2] and GPIOA[3].

CMUX is in the third row from the bottom in Figure 2 and includes GPIOA[4] and GPIOA[5]. And so on.

Therefore, for our example, the I2C master function at GPIO[5:4] is at CMUX and you should use PIN_CMUX for GPIO[5:4].

DA16200 Pin Multiplexing

value / bit sel	0	1	2	3	4	5	6	7	8	9	10
FSEL_GPIO[31:30]	Semi-fixed pin : JTAG_01 TMS TCLK	D_SYS D_SYS_OUT[0] D_SYS_OUT[1]	GPIO(2) x GPIOA[15]	GPIO(2) x GPIOA[15]							
FSEL_GPIO[29:28] FSEL_GPIO[27:26] FSEL_GPIO[25:24]	UART2_01 I2S_CLK_In UART2_RXD UART2_TXD	BT(0:2) BT_sig2 (iBTPri) BT_sig1 (iBtAct) BT_sig0 (oWlanAct)	D_SYS D_SYS_CLK D_SYS_OUT[3] D_SYS_OUT[2]	GPIO(2) GPIOA[14] GPIOA[13] GPIOA[12]							
FSEL_GPIO[22:20]	G(1) + BT GPIOA[11] BT_sig2 (iBTPri)	G(1) + I2S GPIOA[11] I2S_CLK_In	G(1) + eMMC(6) GPIOA[11] mSDeMMC_WRP	sSPI (2:3) sSPI_MOSI sSPI_MISO	UART2 (0:1) UART2_RXD UART2_TXD	mSPI (4:5) E_SPI_IO3 E_SPI_IO2	GPIO(2) GPIOA[11] GPIOA[10]	GPIO(2) GPIOA[11] GPIOA[10]			
FSEL_GPIO[19:16]	5G control(45) 5GC_Sig[5] 5GC_Sig[4]	sSPI (2:3) sSPI_MOSI sSPI_MISO	eMMC (0:1) mSDeMMC_D0 mSDeMMC_D1	sSDIO (0:1) sSDIO_D0 sSDIO_D1	I2C_master mI2C_CLK mI2C_SDA	BT (0:1) BT_sig1 (iBtAct) BT_sig0 (oWlanAct)	mSPI (2:3) E_SPI_IO1 E_SPI_IO0	I2S(0:1) I2S_MCLK I2S_BCLK	GPIO(2) GPIOA[9] GPIOA[8]	GPIO(2) GPIOA[9] GPIOA[8]	
FSEL_GPIO[15:12]		sSPI (0:1) sSPI_CLK sSPI_CSB	eMMC (2:3) mSDeMMCIO_D2 mSDeMMCIO_D3	sSDIO (2:3) sSDIO_D2 sSDIO_D3	UART1 (0:1) UART1_RXD UART1_TXD	I2C slave sI2C_CLK sI2C_SDA	mSPI (0:1) E_SPI_CLK E_SPI_CSB	I2S(2:3) I2S_LRCK I2S_SDO	GPIO(2) GPIOA[7] GPIOA[6]	GPIO(2) GPIOA[7] GPIOA[6]	
FSEL_GPIO[11:8]	5G control(01) 5GC_Sig[3] 5GC_Sig[2]	I2C slave sI2C_CLK sI2C_SDA	eMMC (4:5) mSDeMMC_CLK mSDeMMC_CMD	sSDIO (4:5) sSDIO_CLK sSDIO_CMD	UART1 (2:3) UART1_CTS UART1_RTS	I2C master mI2C_CLK mI2C_SDA	UART1 (0:1) UART1_RXD UART1_TXD	I2S(0:1) I2S_MCLK I2S_BCLK	GPIO(2) GPIOA[5] GPIOA[4]	GPIO(2) GPIOA[5] GPIOA[4]	
FSEL_GPIO[7:4]	AD12 (2) X (Analog In) X (Analog In)	sSPI (0:1) sSPI_CLK sSPI_CSB	I2S (2:3) I2S_LRCK I2S_SDO	I2C slave sI2C_CLK sI2C_SDA	UART1 (0:1) UART1_RXD UART1_TXD		AD12(1) + GPIO(1) GPIO[3] X (Analog In)	AD12(1) + I2S_CLK I2S_CLK_In X (Analog In)	GPIO(2) GPIOA[3] GPIOA[2]	GPIO(2) GPIOA[3] GPIOA[2]	
FSEL_GPIO[3:0]	AD12 (2) X (Analog In) X (Analog In)	sSPI (2:3) sSPI_MOSI sSPI_MISO	I2S (0:1) I2S_MCLK I2S_BCLK	I2C slave sI2C_CLK sI2C_SDA	UART1 (0:1) UART1_RXD UART1_TXD	I2C master mI2C_CLK mI2C_SDA	5G control(01) 5GC_Sig[1] 5GC_Sig[0]	AD12(1) + GPIO(1) GPIO[1] X (Analog In)	AD12(1) + WRP mSDeMMC_WRP X (Analog In)	GPIO(2) GPIOA[1] GPIOA[0]	GPIO(2) GPIO[1] GPIO[0]

Figure 2: Pin Mux for the GPIOA Group

After that, you can select a function to be used with “config” parameter in the API. The <DA16200_ioconfig.h> file defines the possible “config” parameter values.

For our example, when the I2C master is CMUX, the CMUX values defined in the file are the following:

```
// CMUX
#define       CMUX_5GC                    0      /* 5G control[3:2]  */
#define       CMUX_I2Cs                   1      /* I2C slave                */
#define       CMUX_SDm                    2      /* mSDeMMC_CLK,mSDeMMC_CMD */
#define       CMUX_SDs                    3      /* sSDIO_CLK,sSDIO_CMD */
#define       CMUX_UART1c          4      /* UART1(CTS,RTS)    */
#define       CMUX_I2Cm                   5      /* I2C master(SDA,CLK) */
#define       CMUX_UART1d          6      /* UART1(RXD,TXD)    */
#define       CMUX_I2S                    7      /* I2S(MCLK,BCLK)    */
#define       CMUX_GPIO                   8      /* GPIOA[5:4]        */
#define       CMUX_GPIOALT                9      /* GPIOA[5:4]        */
```

For the I2C master, CMUX_I2Cm should be used and its value is 5.

NOTE
In the pin name, *‘m’* at I2Cm means master, while *‘s’* at I2Cs means slave.

In the same way, GPIOC[8:6] can be configured according to Figure 3 or the device API, _fc9k_io_pinmux(UINT32 mux, UINT32 config) with the “mux” value of PIN_UMUX.

NOTE
Only the GPIOC[8:6] pin is available in the GPIOC group in a 6x6 package.

```
// UMUX
#define UMUX_JTAG                   0    /* TDI,TDO,nTRST */
#define UMUX_UART2GPIO              1    /* UART2(TXD,RXD),GPIOC[8] */
#define UMUX_GPIO                   2    /* GPIOC[8:6] */
```

For example, if you want to use UART2 in DA16200, and GPIOC[7:6] are selected for it, then you can call the device API as below:

```
_fc9k_io_pinmux(PIN_UMUX, UMUX_UART2GPIO);
```

NOTE
For the UMUX case, three pins (GPIOC[8:6]) are defined by one single register value of ‘1’ as shown in Figure 3.

DA16200 Pin Multiplexing

value / bit sel	0	1	2	3	4	5
FSEL_GPIO2[21:20]	JTAG(2:3) TDI TDO nTRST	G(1) + UART2 (0:1) GPIO[8] UART2_RXD UART2_TXD	GPIO(2) GPIOC[8] GPIOC[7] GPIOC[6]			
FSEL_GPIO2[19:18]	G(1) + I2S I2S_CLK_In GPIO[4]	UART2 (2:3) UART2_CTS UART2_RTS	GPIO(2) E_SPI_IO3 E_SPI_IO2	GPIO(2) GPIOC[5] GPIOC[4]		
FSEL_GPIO2[17:16]	UART1 (2:3) UART1_CTS UART1_RTS	5G control(45) 5GC_Sig[1] 5GC_Sig[0]	GPIO(2) GPIOC[3] GPIOC[2]			
FSEL_GPIO2[15:14]	UART1 (0:1) UART1_RXD UART1_TXD	I2C master mI2C_CLK mI2C_SDA	GPIO(2) GPIOC[1] GPIOC[0]			
FSEL_GPIO2[13:12]	I2S(0:3) I2S_LRCK I2S_SDO I2S_MCLK I2S_BCLK	QSPI (4:7) H_SPI_DIO7 H_SPI_DIO6 H_SPI_DIO5 H_SPI_DIO4	GPIOB(4) GPIOB[11] GPIOB[10] GPIOB[9] GPIOB[8]			
FSEL_GPIO2[11:10]	mSPI (0:3) E_SPI_IO1 (mSPI_MISO) E_SPI_IO0 (mSPI_MOSI) E_SPI_CSB[0] E_SPI_CLK	5G control(0:3) 5GC_Sig[5] 5GC_Sig[4] 5GC_Sig[3] 5GC_Sig[2]	GPIOB(4) GPIOB[7] GPIOB[6] GPIOB[5] GPIOB[4]	GPIOB(4) GPIOB[7] GPIOB[6] GPIOB[5] GPIOB[4]		
FSEL_GPIO2[9:8]	sSPI(0:3) sSPI_CLK sSPI_CSB sSPI_MOSI sSPI_MISO		GPIOB(4) GPIOB[3] GPIOB[2] GPIOB[1] GPIOB[0]	GPIOB(4) GPIOB[3] GPIOB[2] GPIOB[1] GPIOB[0]		
FSEL_GPIO2[6:4]	QSPI (4:5) F_IO3 (F_HOLD) F_IO2 (F_WP)	UART2 (0:1) UART2_TXD UART2_RXD		sSDIO(4:5) sSDIO_D3 sSDIO_D2	GPIO(2) GPIOC[14] GPIOC[13]	GPIO(2) GPIOC[14] GPIOC[13]
FSEL_GPIO2[3:0]	QSPI (3:0) F_IO1 (F_SI) F_IO0 (F_SO) F_CLK F_CSB[1]	sSPI (3:0) sSPI_MISO sSPI_MOSI sSPI_CLK sSPI_CSB	I2S(0:3) I2S_SDO I2S_LRCK I2S_MCLK I2S_BCLK	sSDIO(0:3) sSDIO_D1 sSDIO_D0 sSDIO_CLK sSDIO_CMD	GPIO(4) GPIOC[12] GPIOC[11] GPIOC[10] GPIOC[9]	GPIO(4) GPIOC[12] GPIOC[11] GPIOC[10] GPIOC[9]

Figure 3: PIN MUX for the GPIOB and GPIOC Groups

3 Register Maps

Table 1: Register Maps for Pin Multiplexing

Address	Registers	Description
Common control for GPIO Pin Status		
0x5000_1208	FSEL_GPIO1	Function Selection of the GPIOA [14:0]
0x5000_120C	FSEL_GPIO2	Function Selection of the GPIOB[11:0] and GPIOC[8:0]

Table 2: FSEL_GPIO1 (0x5000_1208, Reset: 0x3F61_1389)

Bit	Mode	Symbol	Description	Reset
31:30	R/W	-	Pin function selection for GPIOA[15]	b 00
29:28			Pin function selection for GPIOA[14]	b 11
27:26			Pin function selection for GPIOA[13]	b 11
25:24			Pin function selection for GPIOA[12]	b 11
22:20			Pin function selection for GPIOA[11:10]	b 110
19:16			Pin function selection for GPIOA[9:8]	b 0001
15:12			Pin function selection for GPIOA[7:6]	b 0001
11:8			Pin function selection for GPIOA[5:4]	b 0011
7:4			Pin function selection for GPIOA[3:2]	b 1000
3:0			Pin function selection for GPIOA[1:0] See Figure 2	b 1001

Table 3: FSEL_GPIO2 (0x5000_120C, Reset: 0x002E_AA00)

Bit	Mode	Symbol	Description	Reset
21:20	R/W	-	Pin function selection for GPIOC[8:6]	b 01
19:18			Pin function selection for GPIOC[5:4]	b 11
17:16			Pin function selection for GPIOC[3:2]	b 10
15:14			Pin function selection for GPIOC[1:0]	b 11
13:12			Pin function selection for GPIOB[11:8]	b 00
11:10			Pin function selection for GPIOB[7:4]	b 11
9:8			Pin function selection for GPIOB[3:0]	b 00
6:4			Pin function selection for GPIOC[14:13]	b 000
3:0			Pin function selection for GPIOC[12:9]	b 0000
			See Figure 2	

Revision History

Revision	Date	Description
1.1	01-Dec-2020	Changed Figure 1 Register Maps for Pin Mutiplexing.
1.0	30-Oct-2020	Initial version.

DA16200 Pin Multiplexing

Status Definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.

Disclaimer

RoHS Compliance

Dialog Semiconductor's suppliers certify that its products are in compliance with the requirements of Directive 2011/65/EU of the European Parliament on the restriction of the use of certain hazardous substances in electrical and electronic equipment. RoHS certificates from our suppliers are available on request.

Contacting Dialog Semiconductor

United Kingdom (Headquarters)
Dialog Semiconductor (UK) LTD
Phone: +44 1793 757700

Germany
Dialog Semiconductor GmbH
Phone: +49 7021 805-0

The Netherlands
Dialog Semiconductor B.V.
Phone: +31 73 640 8822

Email:
enquiry@diasemi.com

North America
Dialog Semiconductor Inc.
Phone: +1 408 845 8500

Japan
Dialog Semiconductor K. K.
Phone: +81 3 5769 5100

Taiwan
Dialog Semiconductor Taiwan
Phone: +886 281 786 222

Web site:
www.dialog-semiconductor.com

Hong Kong
Dialog Semiconductor Hong Kong
Phone: +852 2607 4271

Korea
Dialog Semiconductor Korea
Phone: +82 2 3469 8200

China (Shenzhen)
Dialog Semiconductor China
Phone: +86 755 2981 3669

China (Shanghai)
Dialog Semiconductor China
Phone: +86 21 5424 9058