Application Note: GT-AN-090623

Extended Register Access for Cirque's Pinnacle ASIC

This is a companion document to
"Interfacing to Cirque's Pinnacle ASIC through SPI or I^2C".

This documnt describes how to access Pinnacle memory beyond the registers accessed with Register Access Protocol (RAP).

This document applies to Cirque's Pinnacle ASIC 2.4

CIRQUE CORPORATION
Input Solutions At Your Fingertips

2463 South 3850 West, Suite A, Salt Lake City, Utah 84120 U.S.A.
Phone: (801) 467-1100 ● Fax: (801) 467-0208 ● www.cirque.com

Table of Contents

Revision History

Date	Previous Revision	Current Revision	Description

1.0 Extended Register Access

Using four standard RAP registers, the host can gain Extended Register Access (ERA) to Pinnacle memory. Register 0x1B is used for the value to be written or read (see Table 1). Register 0x1C is the high byte and register 0x1D is the low byte of the address to be read or written to (see Table 2 and Table 3). Register 0x1E, the ERA control register, specifies READ or WRITE and the optional Auto-Incremented READ or WRITE for sequential commands, as well as a WRITE/Verify option if both the READ and WRITE flags are set (see Table 4). The control register value returns to 0x00 to indicate a command is complete. Use standard Register Access Protocol (RAP) to access these four registers.

It is important to note that accessing the extended registers will assert the command complete (SW_CC) flag and force the hardware data ready (HW_DR) pin high. The Pinnacle data feed should be disabled before accessing extended registers, and SW_CC should be cleared when the host is finished accessing extended registers.

Table 1: Extended Register Access Value - (RAP register 0x1B)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description	Value7	Value6	Value5	Value4	Value3	Value2	Value1	Value0
Read/Write	R/W							
Values	0x00 – 0xFF							
Default	0							

Table 2: Extended Register Access Address High Byte - (RAP register 0x1C)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description	Address15	Address14	Address13	Address12	Address11	Address10	Address9	Address8
Read/Write	R/W							
Values	0x00 – 0xFF							
Default	0							

Table 3: Extended Register Access Address Low Byte - (RAP register 0x1D)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description	Address7	Address6	Address5	Address4	Address3	Address2	Address1	Address0
Read/Write	R/W							
Values	0x00 – 0xFF							
Default	0							

Table 4: Extended Register Access Control - (RAP register 0x1E)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description					WRITE Auto-Increment	READ Auto-Increment	Write	Read
Read/Write					R/W	R/W	R/W	R/W
Values					1=enabled 0=disabled	1=enabled 0=disabled	1=WRITE	1=READ
Default					0	0	0	0

This register value returns to 0x00 to indicate a completed command
Asserting both Bit[1] and Bit[0] indicates a WRITE/Verify

1.1 Extended Register Access Examples

Using standard RAP to send READ and WRITE commands to Pinnacle, the following examples demonstrate the proper sequence to read and write to Pinnacle's extended registers.

Example: READ an Extended Register

1. WRITE the high byte of the 16-bit extended register address to RAP Register 0x1C (ERA High Byte).
2. WRITE the low byte of the 16-bit extended register address to RAP Register 0x1D (ERA Low Byte).
3. WRITE 0x01 (ERA READ flag) to RAP Register 0x1E (ERA Control).
4. READ the RAP Register 0x1E (ERA Control) until it contains 0x00.
5. READ the new value in RAP Register 0x1B (ERA Value).
6. WRITE 0x00 to RAP Register 0x02 (Status1) to clear Command Complete (SW_CC).

Example: READ an Extended Register with Address Increment

1. WRITE the high byte of the 16-bit extended register address to RAP Register 0x1C (ERA High Byte).
2. WRITE the low byte of the 16-bit extended register address to RAP Register 0x1D (ERA Low Byte).
3. WRITE 0x05 to RAP Register 0x1E (ERA Control) to specify auto-increment read.

Repeat 4, 5, and 6 as needed

4. READ the RAP Register 0x1E (ERA Control) until it contains 0x00.
5. READ the new value in RAP Register 0x1B (ERA Value).
6. WRITE 0x00 to RAP Register 0x02 (Status1) to clear Command Complete (SW_CC).

Extended Register Address is incremented.
Repeat steps 4, 5, and 6 to reach the desired address.

Example: WRITE to an Extended Register

1. WRITE the value to be written in RAP Register 0x1B (ERA Value)
2. WRITE the high byte of the 16-bit extended register address to RAP Register 0x1C (ERA High Byte).
3. WRITE the low byte of the 16-bit extended register address to RAP Register 0x1D (ERA Low Byte).
4. WRITE 0x02 (ERA WRITE flag) to RAP Register 0x1E (ERA Control).
5. READ the RAP Register 0x1E (ERA Control) until it contains 0x00.
6. WRITE 0x00 to RAP Register 0x02 (Status1) to clear Command Complete (SW_CC).

Example: WRITE to an Extended Register with Address Increment

1. WRITE the value to be written in RAP Register 0x1B (ERA Value)
2. WRITE the high byte of the 16-bit extended register address to RAP Register 0x1C (ERA High Byte).
3. WRITE the low byte of the 16-bit extended register address to RAP Register 0x1D (ERA Low Byte).
4. WRITE 0x0A (ERA auto-increment WRITE) to RAP Register 0x1E (ERA Control).

Repeat Steps 5 and 6 as needed

5. READ the RAP Register 0x1E (ERA Control) until it contains 0x00.
6. WRITE 0x00 to RAP Register 0x02 (Status1) to clear Command Complete (SW_CC).

Extended Register Address is incremented.
Repeat steps 5 and 6 to reach the desired address.

Example: WRITE to an Extended Register with Verification

1. WRITE the value to be written in RAP Register 0x1B (ERA Value)
2. WRITE the high byte of the 16-bit extended register address to RAP Register 0x1C (ERA High Byte).
3. WRITE the low byte of the 16-bit extended register address to RAP Register 0x1D (ERA Low Byte).
4. WRITE 0x03 (write and read) to RAP Register 0x1E (ERA Control).
5. READ the RAP Register 0x1E (ERA Control) until it contains 0x00.
6. READ the value in RAP Register 0x1B (ERA Value) to verify.
7. WRITE 0x00 to RAP Register 0x02 (Status1) to clear Command Complete (SW_CC).

1.2 Sample Code

2.1.1 ERA READ

```
/*---------------------------------------------------------------------------*\

    void PinnacleIOExt_ReadMemory( unsigned int16 StartAddress,
                                   unsigned int16 NumBytes,
                                   unsigned int8* pValueBuffer )

\*---------------------------------------------------------------------------*/
void PinnacleIOExt_ReadMemory( unsigned int16 StartAddress, unsigned int16 NumBytes, unsigned
    int8* pValueBuffer )
{
    unsigned int16 temp16 = 0;
    unsigned int8 val;
    unsigned int16 CurrentAddress;

    CurrentAddress = StartAddress;

    // ---> All addresses in lower 32 regs (0x00-0x1F) can be read directly
    while( CurrentAddress <= HOSTREG__31 )
    {
        PinnacleIO_ReadRegister((unsigned int8)(CurrentAddress & 0x00FF),pValueBuffer + temp16 );
        CurrentAddress++;
        temp16++;

        if( temp16 == NumBytes )
            return;
    }

    // ---> All addresses > 0x1F must be read indirectly

    // write 16 bit address to extended address high/low regs
    PinnacleIO_WriteRegister(HOSTREG__EXT_REG_AXS_ADDR_HIGH,(unsigned int8)(CurrentAddress >> 8));
    PinnacleIO_WriteRegister(HOSTREG__EXT_REG_AXS_ADDR_LOW,(unsigned int8)(CurrentAddress & 0x00FF));

    // loop through remaining byte
    for( ; temp16 < NumBytes; temp16++ )
    {
        //write ERA control register to force extended read and post-increment extended address
       PinnacleIO_WriteRegister(HOSTREG__EXT_REG_AXS_CTRL, HOSTREG__EREG_AXS__READ |
         HOSTREG__EREG_AXS__INC_ADDR_READ );

        //loop reading ERA control register until value is 0x00,indicating write is complete)
        do
        {
            PinnacleIO_ReadRegister( HOSTREG__EXT_REG_AXS_CTRL, &val );

        } while( val != 0x00 );

        // read extended val reg which now contains value at extended address
        PinnacleIO_ReadRegister( HOSTREG__EXT_REG_AXS_VALUE, pValueBuffer + temp16  );
    }
    return;
}
```

2.1.2 ERA WRITE

```
/*-------------------------------------------------------------------------*\

    void PinnacleIOExt_WriteMemory( unsigned int16 StartAddress,
                                    unsigned int16 NumBytes,
                                    unsigned int8* pValueBuffer )

\*-------------------------------------------------------------------------*/

void PinnacleIOExt_WriteMemory(unsigned int16 StartAddress, unsigned int16 NumBytes, unsigned
    int8* pValueBuffer )
{
    unsigned int16 temp16 = 0;
    unsigned int8 val;
    unsigned int16 CurrentAddress;

    CurrentAddress = StartAddress;

    // ---> All addresses in lower 32 regs (0x00-0x1F) can be written directly
    while( CurrentAddress <= HOSTREG__31 )
    {
        PinnacleIO_WriteRegister((unsigned int8)(CurrentAddress & 0x00FF), *(pValueBuffer + temp16));
        CurrentAddress++;
        temp16++;

        if( temp16 == NumBytes )
            return;
    }

    // ---> All addresses > 0x1F must be written indirectly

    // write 16 bit address to extended address high/low regs
    PinnacleIO_WriteRegister(HOSTREG__EXT_REG_AXS_ADDR_HIGH,(unsigned int8)(CurrentAddress >> 8) );
    PinnacleIO_WriteRegister(HOSTREG__EXT_REG_AXS_ADDR_LOW,(unsigned int8)(CurrentAddress & 0x00FF));

    // loop through remaining byte
    for( ; temp16 < NumBytes; temp16++ )
    {
        // write 8 bit value to be written to extended value reg
        PinnacleIO_WriteRegister( HOSTREG__EXT_REG_AXS_VALUE, *(pValueBuffer + temp16) );

        // write ERA control register to force extended write and post-increment extended address
        PinnacleIO_WriteRegister( HOSTREG__EXT_REG_AXS_CTRL, HOSTREG__EREG_AXS__WRITE |
          HOSTREG__EREG_AXS__INC_ADDR_WRITE );

        // loop reading ERA control register until value is 0x00 indicating write is complete)
        do
        {
            PinnacleIO_ReadRegister( HOSTREG__EXT_REG_AXS_CTRL, &val );
        } while( val != 0x00 );
    }
    return;
}
```