Application Note: GT-AN-090625

Using a Stylus with Cirque's Pinnacle ASIC

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

This document describes how to use a tethered stylus with Cirque's Pinnacle ASIC.

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 Using a Stylus with Pinnacle

Cirque's Pinnacle ASIC not only senses a finger, but allows touch input by way of a tethered stylus. Using the stylus allows for higher sample rates and a precise tip which can accommodate functions such as signature capture. This document describes the firmware functions involved in using Pinnacle, with a standard 5.25 mm stylus or a small 2 mm stylus, at 200 or 300 samples per second.

1.1 Registers required for Stylus Functionality

Stylus functionality requires modification to the Sample Rate register in Pinnacle's standard register set and to extended registers shown in Table 1. The protocol for extended register access (ERA) is explained in "*GT-AN-090623 Extended Register Access for Cirque's Pinnacle ASIC*". The format for each register is shown in Table 4 through Table 7.

Table 1: Registers used for Stylus Functionality

Register Set	Address	Function	Comment
Standard	0x06	FeedConfig3	This is a standard RAP register
Standard	0x09	Sample Rate	This is a standard RAP register
Extended	0x00EB	ADC_MuxControl	This register controls the ADC mux
Extended	0x019F	Packet Timer Reload	This sets the packet timer for timing the sample rate
Extended	0x019E	Track Timer Reload	This times the tracking, and should be the same as the packet timer
Extended	0x0187	Tracking ADC Config	This sets advanced settings for the ADC
Extended	0x01DF	Comp Values	This is the starting address of the comp values

Table 2: FeedConfig3 - Register 0x06

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description	SW EMI detection Disable[1]	HW EMI detection Disable[2]	Dynamic EMI adjust Disable[3]	WRAP lockout Disable[4]	Noise Avoidance Disable[5]	Palm/NERD measurements Disable[6]	Smoothing Disable[7]	DualPoint Buttons[8]
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Values	-	-	-	-	1=disable 0=enable	1=disable 0=enable	1=disable 0=enable	1=enable 0=disable
Default	0	0	0	0	0	0	0	0

See Appendix A for notes associated with this register.

Table 3: Sample Rate (number of feed samples created per second) – Register 0x09.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description	B7	B6	B5	B4	B3	B2	B1	B0
Read/Write	R/W							
Values	0x00 (turns sample rate control to the track timer and packet reload timers) 0x64 (100 samples/second) 0x50 (80 samples/second) 0x3C (60 samples/second) 0x28 (40 samples/second) 0x14 (20 samples/second) 0x0A (10 samples/second) Writing any other value will default to 0x64 (100 samples/second)							
Default	0x64 (100 samples/second)							

Table 4: ADC Mux Control

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description						SNSN Enable		SNSP Enable
Read/Write						R/W		R/W
Values						1=stylus 0=no stylus		1=enabled 0=disable
Default						0		0

1. SNSP is the ADC input that is used to measure the finger. Asserting this flag will enable finger operation.
2. SNSN is the ADC input that is used for stylus operation. Asserting this flag will enable stylus operation.

Table 5: Packet Timer Reload

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description	B7	B6	B5	B4	B3	B2	B1	B0
Read/Write	R/W							
Values	0x06 (300 samples/second) 0x09 (200 samples/second) 0x13 (100 samples/second) Contents should match that of the Track Timer.							
Default	0x13 (100 samples/second)							

Table 6: Track Timer Reload

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description	B7	B6	B5	B4	B3	B2	B1	B0
Read/Write	R/W							
Values	0x06 (300 samples/second) 0x09 (200 samples/second) 0x13 (100 samples/second) Contents should match that of the Packet Timer.							
Default	0x13 (100 samples/second)							

Table 7: Tracking ADC Config

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description	ADC Gain	ADC Gain						
Read/Write	R/W							
Values	8 byte Value (Bits 7 & 6) 0xC0 = 4x gain 0x80 = 3x gain 0x40 = 2x gain 0x00 = 1x gain							
Default	0							

1.2 Changing Sample Rate: 100 sps, 200 sps, 300 sps

Pinnacle defaults to 100 samples per second (sps). In order to change the sample rate, write 0x00 to Register 0x09, Sample Rate (see Table 3). Control will be turned to the Packet Timer and the sample rate can then be set up to 300 samples per second by writing the desired value to Registers 0x01DE and 0x01DF, Packet Timer Reload and Track Timer Reload (see Table 5 and Table 6).

Sample rates higher than 100 sps require that noise avoidance and NERD measurements be disabled in Register 0x06 (FeedConfig3) to accommodate the higher rate (see Table 2). The example code below shows the proper sequences for setting the sample rate to 100, 200 and 300 sps.

2.1.1 Sample Code: Changing sample rate by writing 100, 200 and 300 sps

```
unsigned int8 Temp;
PinnacleIO_WriteRegister(HOSTREG_SAMPLERATE, 0x00);


//100sps//
//Step 1–correct settings for the noise avoidance, etc
PinnacleIO_ReadRegister(HOSTREG_FEEDCONFIG3,&Temp);
Temp &= ~(HOSTREG_FEEDCONFIG3_DISABLE_NOISE_AVOIDANCE|HOSTREG_FEEDCONFIG3_DISABLE_PALM_NERD_MEAS);
PinnacleIO_WriteRegister(HOSTREG_FEEDCONFIG3, &Temp);

//Step 2–correct timing for 100sps
Temp = 0x13;
PinnacleIO_WriteRegister(PDATA_PacketTimerReload,&Temp);
PinnacleIO_WriteRegister(PDATA_TrackTimerReload,&Temp);



//200sps//
//Step 1–correct settings for the noise avoidance, etc
PinnacleIO_ReadRegister(HOSTREG_FEEDCONFIG3,&Temp);
Temp |= (HOSTREG_FEEDCONFIG3_DISABLE_NOISE_AVOIDANCE|HOSTREG_FEEDCONFIG3_DISABLE_PALM_NERD_MEAS);
PinnacleIO_WriteRegister(HOSTREG_FEEDCONFIG3, &Temp);

//Step 2–correct timing for 200sps
Temp = 0x09;
PinnacleIO_WriteRegister(PDATA_PacketTimerReload,&Temp);
PinnacleIO_WriteRegister(PDATA_TrackTimerReload,&Temp);



//300sps//
//Step 1–correct settings for the noise avoidance, etc
PinnacleIO_ReadRegister(HOSTREG_FEEDCONFIG3,&Temp);
Temp |= (HOSTREG_FEEDCONFIG3_DISABLE_NOISE_AVOIDANCE|HOSTREG_FEEDCONFIG3_DISABLE_PALM_NERD_MEAS);
PinnacleIO_WriteRegister(HOSTREG_FEEDCONFIG3, &Temp);

//Step 2–correct timing for 300sps
Temp = 0x06;
PinnacleIO_WriteRegister(PDATA_PacketTimerReload,&Temp);
PinnacleIO_WriteRegister(PDATA_TrackTimerReload,&Temp);
```

1.3 Input Modes: Pen only, Finger only, Pen and Finger

Input to Pinnacle is controlled by the ADC_MuxControl register at address 0x00EB. Writing 0x01 to that register enables finger input. Writing 0x04 enables pen input. Writing 0x05 enables both pen and finger input. Anytime that register is written to it should be done so that none of the upper bits are modified.

EXAMPLE: Read/Modify/Write

Enable pen only in Register 0x00EB (ADC_MuxControl)

```
unsigned int8 Temp;
PinnacleIO_ReadRegister(ADC_MuxControl, &Temp);
Temp |= 0x04;
PinnacleIO_WriteRegister(ADC_MuxControl, &Temp);
```

1.4 Auto-detecting stylus or finger input

Auto-detecting between a finger or the stylus as the input allows for simultaneous operation of pen and finger when using a small tip. The difference between a finger and the small 2 mm pen tip requires advanced settings. With the larger 5.25 mm pen tip, no advanced settings are required to accommodate the pen.

Auto-detecting the input method (pen or finger) also allows Z level thresholds and special filters to be applied that can enhance functionality for which ever input type is being used in the moment.

Auto-detection is done in three steps.

1. The first state is a raw tracking mode with settings that track both pen and finger, but are not ideal for either.
2. The internal input mode is then switched from pen + finger to pen only mode. The input is checked again to see if the signal is still there. If the signal still exists, it is concluded that it was caused by the pen and tracking proceeds with the pen specific settings.

   In pen only mode, the sample rate is automatically set to 300 sps. Therefore, detecting a pen is quicker than detecting a finger. By default, all pen input will be at 300 sps.
3. Conversely, if the signal disappears when switching to pen only mode, it is concluded that the input was caused by a finger and the finger specific settings are used.

1.5 Compensation

Compensation/Calibration is an overall capacitance measurement of the sensor and its environment. It is the bare sensor capacitance without any input present on the sensor. With finger input enabled, the calibration data represents the capacitance of the entire sensor. With pen only enabled, the calibration data only represents the capacitance of the pen and its routing on the PCB where the Pinnacle ASIC resides. The two sets of data are significantly different.

When using the small pen tip, several parameters (i.e., Gain, SNSP, SNSN) change which will cause the compensation values to change as well. When switching between input modes, the correct compensation values need to be loaded into Pinnacle's RAM to allow for proper operation.

A Factory Compensation should be performed for finger and stylus input. Each set of factory compensation data should be stored permanently for the Host to load back in to Pinnacle memory during normal use as needed. See "*GT-AN-090623 Extended Register Access for Cirque's Pinnacle ASIC*" and "*GT-AN-090624 Managing Sensor Compensation with Cirque's Pinnacle ASIC*" for more details.

Sample code is provided in the associated files: *SmallPenTracking.c*, *SmallPenTracking.h, and Pinnacle_07_2D_MemMap.h*. The sample code demonstrates auto detection, including calibration, and the correct settings for pen, finger, and raw search mode.