Getting Started With PSoC® 1

Author: Robert Murphy

Associated Project: Yes

Associated Part Family: All PSoC 1 Families

Software Version: PSoC Designer™ 5.4 and higher

Related Application Notes: For a complete list of the application notes, click here.

To get the latest version of this application note, or the associated project file, please visit http://www.cypress.com/go/AN75320.

AN75320 introduces you to PSoC® 1, an 8-bit processor with programmable digital and analog blocks that enable implementation of custom functions. This application note describes the PSoC 1 architecture and development tools, and shows how to create your first design. This document guides you to more resources to accelerate in-depth learning about PSoC 1.

Contents

Introduction

PSoC 1 is a true **programmable embedded system-on-chip**, integrating custom analog and digital peripheral functions, memory and an M8C microcontroller on a single chip.

This is different from most mixed-signal embedded systems, which use a combination of a microcontroller (MCU) and external analog and digital peripherals. This type of system typically requires many integrated circuits in addition to the MCU, such as opamps, ADCs, and digital ASICs.

PSoC 1 provides a low-cost alternative to the combination of MCU and external ICs. In addition to reducing overall system cost, the programmable analog and digital subsystems allow great flexibility, in-field tuning of the design, and accelerates time to market.

The system resources of PSoC 1 can be dynamically reconfigured during run time to perform completely different function. Dynamic reconfigurability is possible during instances where the resources have been consumed in a design. This allows you to reuse PSoC 1 system resources and maximize the integration value proposition.

The capacitive touch-sensing feature in PSoC 1, known as CapSense®, offers unprecedented signal-to-noise ratio, best-in-class waterproofing, and a wide variety of sensor types such as buttons, sliders, track pads, and proximity sensors.

This application note helps you explore the PSoC 1 architecture and development tools, and learn how to create your first project using PSoC Designer™, the development tool for PSoC 1. This document also guides you to more resources to accelerate in-depth learning about PSoC 1.

In addition to PSoC 1, the Cypress PSoC portfolio also contains PSoC 3, PSoC 4, and PSoC 5LP devices. These PSoC devices offer different system architecture and peripherals. For more information, refer to the Cypress Platform PSoC Solutions Roadmap.

Comparison of PSoC 1 Families

PSoC 1 includes thirteen device families. Table 1 shows the features available in these device families.

Table 1. PSoC 1 Device Selector Summary Table

Features	Device Family											
	29x66	28xxx	27x43	24x94	24x93	24x33	24x23A	23x33	22x45	21x45	21x34	21x23
Flash (KB)	32K	16K	16K	16K	32K	8K	4K	8K	16K	8K	8K	4K
SRAM (KB)	2K	1K	256B	512B	2K	256B	256B	256B	1K	512B	512B	256B
ADC 1 (DS/SS)	14-bit (DS)	4x14-bit (DS)	11-bit (DS)	14-bit (DS)	10-bit (INC)	11-bit (DS)	11-bit (DS)	11-bit (DS)	8-bit (SS)	8-bit (SS)	10-bit (SS)	10-bit (SS)
ADC 2 (SAR)	-	10-bit (150 Ksps)	-	-	-	8-bit (300 Ksps)	-	8-bit (300 Ksps)	10-bit (150 Ksps)	10-bit (150 Ksps)	-	-
Comparator	4	4	4	2	2	2	2	2	4	4	2	2
DAC (8-bit)	4	4	4	2	-	2	2	2	-	-	-	-
PGA (x48 Gain)	4	4	4	2	-	2	2	2	-	-	-	-
TCPWM (16-bit)	8	6	4	2	3 (Timer)	2	2	2	4	2	2	2
UART/SPI	4	3	2	1	1 SPI	1	1	1	2	1	1	1
I2C	1	2	1	1	1 (Slave)	1	1	1	1	1	1	1
CapSense Buttons	-	43	-	49	-	-	-	-	37	23	23	-
GPIO	64	44	44	50	36	26	24	26	38	24	24	16
USB	-	-	-	FS 2.0	FS 2.0	-	-	-	-	-	-	-
ECO	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	N
SMP	Y	Y	Y	N	N	N	Y	N	Y	Y	Y	Y
MAC	2	2	1	2	0	1	1	1	1	1	0	0

FS = Full Speed, **SMP** = Switch Mode Pump, **ECO** = External Crystal Oscillator, **MAC** = Multiply Accumulate, **INC** = Incremental, **PGA** = Programmable Gain Amplifier, **TCPWM** = Timer Counter Pulse Width Modulator, **UART** = Universal Asynchronous Receiver Transmitter, **SPI** = Serial Peripheral Interface, **I2C** = Inter-Integrated Circuit

PSoC 1 Feature Set

PSoC 1 has a large set of capabilities and features, which include a CPU core, a digital subsystem, an analog subsystem, a memory subsystem, and system resources, as Figure 1 shows. The following sections give brief descriptions of each feature. For more information, see the PSoC 1 family device datasheet, technical reference manual (TRM), and application notes listed in PSoC 1 Learning Resources.

Figure 1. PSoC 1 Architecture (CY8C29466)

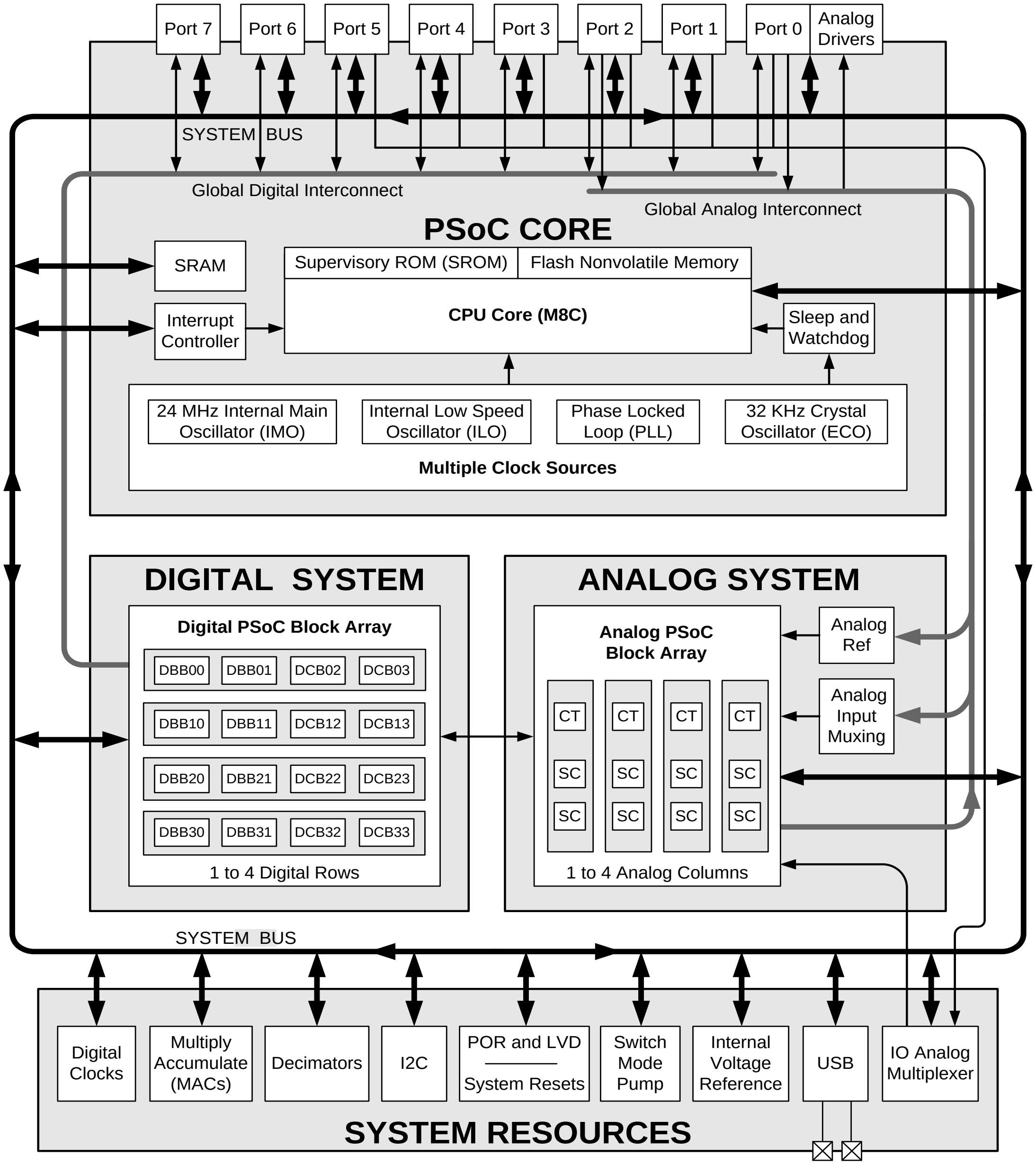


Figure 1 shows the feature available in the CY8C29466 device. Subsets of these features are available in other devices; see Table 1.

M8C Processor and Memory

PSoC 1 has an M8C Processor, which is an 8-bit RISC CPU with Harvard architecture. It is capable of operating at a maximum frequency of 24 MHz, providing 4 MIPS performance. The M8C has 37 instructions. PSoC 1 devices have as much as 2 KB of SRAM, as Table 1 shows. For more details, refer to the PSoC 1 Technical Reference Manual.

Programmable Digital Subsystem

The digital subsystem of PSoC is unique because of its programmability and routing systems. Table 1 lists the various PSoC 1 parts and the digital resources available in each device. The digital blocks in PSoC 1 are sub-divided into Digital Basic Blocks (DBBs) and Digital Communication Blocks (DCBs). The number of DBBs and DCBs are always equal.

Digital Basic Blocks (DBBs)

DBBs are basic configurable digital resources, which you can program to function as timers, counters, or PWMs. Each DBB allows you to place an 8-bit resource. Creating a 16-bit, a 24-bit, or a 32-bit digital resource requires two, three, or four DBBs, respectively.

Digital Communication Blocks (DCBs)

DCBs not only allow you to place basic digital resources (timer, counter, or PWM), but they also allow you to place communication resources, such as SPI and UARTs. You can place DBB components in DCBs. However, you cannot place DCB resources in DBBs.

Programmable Analog Subsystem

The unique analog subsystem of PSoC 1 subsystem is composed of analog blocks arranged in a column configuration. These analog blocks are either continuous time (CT) or switch capacitor (SC) blocks.

Continuous Time (CT) Blocks

The CT blocks inside PSoC are programmable analog blocks that you can configure as either comparators or programmable gain amplifiers (PGAs). The CT blocks are built around low-noise and low-offset opamps. The large number of analog multiplexers in the CT block provides high configurability. For more information on the structure and composition of CT blocks, refer to Chapter 22 of the PSoC 1 Technical Reference Manual.

Switched Capacitor (SC) Blocks

SC blocks are also built around low-noise, low-offset opamps surrounded by analog multiplexers. These blocks are unique because groups of capacitors and switches surround the opamps and multiplexers. There is no resistor array as there are in the CT blocks. For additional information on SC block architecture, refer to Chapter 23 of the PSoC 1 Technical Reference Manual or AN2041 – Understanding PSoC 1 Switch Capacitor Analog Blocks.

System Wide Resources

Clocking System

PSoC includes an advanced clocking system with multiple clock sources, many of which are programmable. You can derive the main clock source from either the internal 24-MHz internal main oscillator (IMO) or an external clock source of up to 24 MHz. In addition, for a low-speed oscillator, you can use either a 32-kHz oscillator circuit or an internal low-speed oscillator (ILO). For more information on clocks available in PSoC 1, refer to AN32200 - PSoC® 1 - Clocks and Global Resources.

Dynamically Reconfigurability

With PSoC 1, you can start a project design without knowing all the details of the end product, and then add resources as needed. As long as the digital or analog resources are available, you can adjust, adapt, and grow. Even in instances where the resources have been consumed in a design, PSoC can be dynamically reconfigured during run time to perform a different function. This allows you to reuse resources in PSoC and maximize the integration value proposition. For details, refer to AN2104 - PSoC®1 - Dynamic Reconfiguration With PSoC® Designer™.

Switch Mode Pump (SMP)

The switch mode pump (SMP) is a DC/DC boost circuit supported by PSoC to allow the device to operate from a single 1.5-V battery. PSoC has an internal FET and an independent PWM hardware to run the boost circuit. You need only an external battery, inductor, diode, and capacitor to complete the boost circuitry. For more information, refer to Chapter 30 of PSoC 1 Technical Reference Manual.

Multiply Accumulate (MAC)

The Multiply Accumulate, or MAC, provides an 8-bit signed multiplier with a 32-bit accumulator for summation. The MAC is extremely useful in performing math operations and implementing digital filters. You can use the MAC by writing to and reading from certain registers of the device. After multiplication, you can either read the value out of the register or store it in the accumulator. You can clear the accumulator reset it to a value of zero by writing to the clear registers (MACx_CL1 and MACx_CL2). For more information, refer to Chapter 26 of PSoC 1 Technical Reference Manual.

Voltage Reference

PSoC 1 contains many options for voltage references. There are three main terms are:

(i) AGND (ii) RefHi (iii) RefLo

Analog signals in the device are biased to analog ground (AGND). The voltage location of AGND depends on the developer, who has a range of options. Analog signals higher than AGND are considered positive, while voltages below it are considered negative. RefHi and RefLo refer to the upper and lower limits of the analog system.

Dedicated I^{2}C Hardware

The I^{2}C communications block is a serial-to-parallel processor, designed to interface the PSoC device to a two-wire I^{2}C serial communications bus. To eliminate the need for excessive M8C microcontroller intervention and overhead, the block has I^{2}C-specific support for status detection and generation of framing bits.

Following are the major features and capabilities of the PSoC I^{2}C hardware controller:

- Industry-standard Philips® I^{2}C bus-compatible interface
- Master and slave operation, multi-master capable
- Only two pins (SDA and SCL) are required to interface to the I^{2}C bus
- Standard data rate of 100/400 kbps; also supports 50 kbps
- 7-bit addressing mode; 10-bit addressing supported

Additional information on using I^{2}C can be found in AN50987: Getting Started with I2C in PSoC 1.

GPIO System

The GPIO system provides an interface between the CPU, the peripherals and the outside world. Each GPIO, when used as a digital I/O, can source 10 mA per pin and can sink 25 mA per pin. In total, the device is capable of sinking 200 mA, with 100 mA per side. Therefore, even number pins on any port can handle a total of 100 mA, and odd number pins on any port can handle an additional 100 mA. The device can source a total of 80 mA, or 40 mA per side. For more details, refer to Chapter 6 of PSoC 1 Technical Reference Manual.

CapSense

Some PSoC 1 devices support capacitive touch sensing known as CapSense. Capacitive touch sensors are user-interface devices that use human body capacitance to detect the presence of a finger on or near (proximity) a sensor. Capacitive sensors are aesthetically superior, easy-to-use, and have long lifetimes. CapSense allows you to replace expensive and unreliable mechanical buttons with capacitive buttons that are simple copper traces on the PCB. CapSense supports a wide variety of sensors such as buttons, sliders, track pads and proximity sensors. To see which PSoC 1 devices support CapSense, see Table 1. For additional information, refer to the “Getting Started with CapSense” design guide.

Development Tools

Cypress supports PSoC 1 with high-quality development tools such as the PSoC Designer software tool, hardware tools including development kits, programming hardware, and debugging hardware. These tools help you to configure the device, develop the application code, then build, debug, and deploy an embedded design.

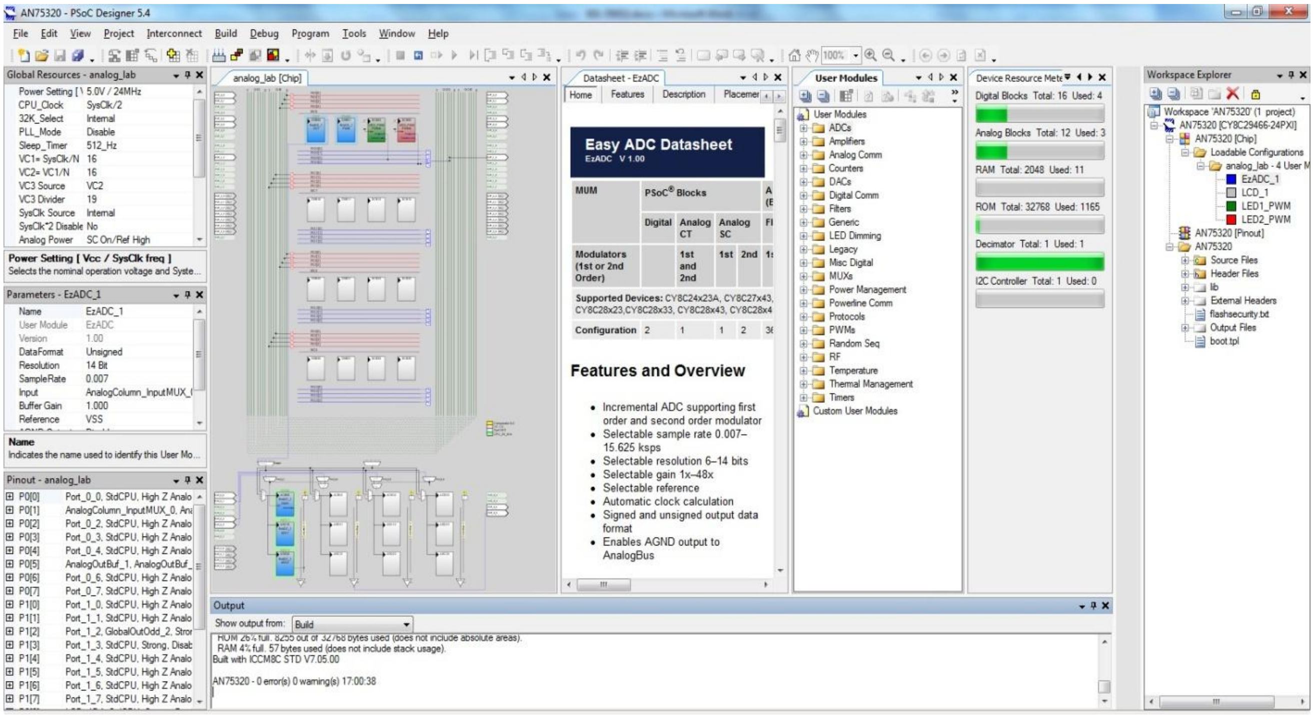
Software: PSoC Designer IDE

Cypress PSoC Designer is an integrated development environment (IDE) used to customize, configure, and program PSoC 1 devices. You can download this tool from http://www.cypress.com/go/psocdesigner.

PSoC Designer is a fully-contained environment where you can create your PSoC application by configuring the analog and digital peripherals, write application code, and perform other functions discussed in this application note. In addition, this software lets you program a PSoC device and debug a project using the PSoC ICE-Cube debug platform.

Figure 2 shows the layout of PSoC Designer and a description of each section of the IDE. Note that this is not the default layout of PSoC Designer. Any windows seen in the following figure that do not appear when you open Designer can be added by navigating through the **View** dropdown menu in the toolbar.

Figure 2. PSoC Designer Layout


AN75320 - PSoC Designer 5.4
File Edit View Project Interconnect Build Debug Program Tools Window Help
100%
Global Resources - analog_lab
Parameters - EzADC_1
Pinout - analog_lab
analog_lab [Chip]
Datasheet - EzADC
Home Features Description Placemer
User Modules
Device Resource Mete
Workspace Explorer
Output

Global Resources - analog_lab

Power Setting [	5.0V / 24MHz
CPU_Clock	SysClk/2
32K_Select	Internal
PLL_Mode	Disable
Sleep_Timer	512_Hz
VC1= SysClk/N	16
VC2= VC1/N	16
VC3 Source	VC2
VC3 Divider	19
SysClk Source	Internal
SysClk*2 Disable	No
Analog Power	SC On/Ref High

Power Setting [Vcc / SysClk freq]
Selects the nominal operation voltage and Syste...

Parameters - EzADC_1

Name	EzADC_1
User Module	EzADC
Version	1.00
DataFormat	Unsigned
Resolution	14 Bit
SampleRate	0.007
Input	AnalogColumn_InputMUX_(
Buffer Gain	1.000
Reference	VSS

Name
Indicates the name used to identify this User Mo...

Pinout - analog_lab

P0[0]	Port_0_0, StdCPU, High Z Analo
P0[1]	AnalogColumn_InputMUX_0, Ana
P0[2]	Port_0_2, StdCPU, High Z Analo
P0[3]	Port_0_3, StdCPU, High Z Analo
P0[4]	Port_0_4, StdCPU, High Z Analo
P0[5]	AnalogOutBuf_1, AnalogOutBuf_
P0[6]	Port_0_6, StdCPU, High Z Analo
P0[7]	Port_0_7, StdCPU, High Z Analo
P1[0]	Port_1_0, StdCPU, High Z Analo
P1[1]	Port_1_1, StdCPU, High Z Analo
P1[2]	Port_1_2, GlobalOutOdd_2, Stror
P1[3]	Port_1_3, StdCPU, Strong, Disab
P1[4]	Port_1_4, StdCPU, High Z Analo
P1[5]	Port_1_5, StdCPU, High Z Analo
P1[6]	Port_1_6, StdCPU, High Z Analo
P1[7]	Port_1_7, StdCPU, High Z Analo

Easy ADC Datasheet
EzADC V 1.00

MUM	PSoC® Blocks				A (E
	Digital	Analog CT	Analog SC		Fl
Modulators (1st or 2nd Order)		1st and 2nd	1st	2nd	1s
Supported Devices: CY8C24x23A, CY8C27x43, CY8C28x23,CY8C28x33, CY8C28x43, CY8C28x4					
Configuration	2	1	1	2	3(

Features and Overview

- Incremental ADC supporting first order and second order modulator
- Selectable sample rate 0.007–15.625 ksps
- Selectable resolution 6–14 bits
- Selectable gain 1x–48x
- Selectable reference
- Automatic clock calculation
- Signed and unsigned output data format
- Enables AGND output to AnalogBus

User Modules
- ADCs
- Amplifiers
- Analog Comm
- Counters
- DACs
- Digital Comm
- Filters
- Generic
- LED Dimming
- Legacy
- Misc Digital
- MUXs
- Power Management
- Powerline Comm
- Protocols
- PWMs
- Random Seq
- RF
- Temperature
- Thermal Management
- Timers

Custom User Modules

Digital Blocks Total: 16 Used: 4
Analog Blocks Total: 12 Used: 3
RAM Total: 2048 Used: 11
ROM Total: 32768 Used: 1165
Decimator Total: 1 Used: 1
I2C Controller Total: 1 Used: 0

Workspace 'AN75320' (1 project)
- AN75320 [CY8C29466-24PXI]
  - AN75320 [Chip]
    - Loadable Configurations
      - analog_lab - 4 User M
        - EzADC_1
        - LCD_1
        - LED1_PWM
        - LED2_PWM
  - AN75320 [Pinout]
  - AN75320
    - Source Files
    - Header Files
    - lib
    - External Headers
    - flashsecurity.txt
    - Output Files
    - boot.tpl

Show output from: Build

```
  ROM 26% full. 8255 out of 32768 bytes used (does not include absolute areas).
  RAM 4% full. 57 bytes used (does not include stack usage).
Built with ICCM8C STD V7.05.00

AN75320 - 0 error(s) 0 warning(s) 17:00:38
```

Hardware

Cypress provides various PSoC 1 hardware kits to fulfill your design requirements. To choose your PSoC 1 device, refer to PSoC 1 Kits and PSoC 1 Kit Selector Guide. The following section provides a list of the development kits, the programming hardware and the debugging hardware.

Development Kits

Cypress provides several development kits to help you easily prototype your PSoC 1 design. Table 2 lists the some of the development kits that support the evaluation of PSoC 1. For complete list of the development kits/board, refer to Development Kits.

Table 2. PSoC 1 Development Kits

Development Kit	Description
CY3210-PSOCEVAL1	The CY3210 PSoC Eval1 Kit enables you to evaluate and experiment Cypress's PSoC 1 programmable system-on-chip design methodology and architecture
CY8CKIT-001	The CY8CKIT-001 PSoC Development Kit provides a common development platform where you can prototype and evaluate different solutions.

Programming Hardware

You can choose from several options for programming devices from Cypress-provided and third-party tools. Cypress offers four programming devices: MiniProg1, MiniProg3, ICE-Cube, and CY3207-ISSP.

MiniProg1

The MiniProg1 is the ISSP programmer that is included with many PSoC 1 kits. You can use this programmer for all PSoC 1 devices, except the CY25/26xxx devices. The MiniProg1 cannot be used to program PSoC 3 or PSoC 5 devices.

MiniProg3

The MiniProg3 is the ISSP programmer that comes with the CY8CKIT-001 Development Kit. The MiniProg3 is an all-in-one programmer for PSoC 1, PSoC 3, and PSoC 5 devices, a debug tool for PSoC 3 and PSoC 5 architectures, and a USB-I2C bridge for debugging I2C serial connections and communicating with PSoC devices.

Third-Party Programmers

See the list of third-party programmer tools at http://www.cypress.com/?rID=2543. These tools have been designed, tested, and qualified by Cypress to support programming of PSoC 1 devices.

Debugging Hardware

CY3215A-DK

CY3215A-DK In-Circuit Emulation Lite Development Kit includes an in-circuit emulator (ICE). While the ICE-Cube is primarily used to debug PSoC 1 devices, it can also program PSoC 1 devices using ISSP. Rather than using the blue CAT5 cable or the flex cable to interface with debug pods, you can use the yellow ISSP cable to program devices. This makes the ICE-Cube useful for both debug and production environments. See AN73212 – Debugging with PSoC 1 for more details.

PSoC 1 Learning Resources

This section provides a list of PSoC 1 learning resources to help you to get started and to develop complete applications with PSoC 1. Many resources are available for PSoC 1 developers, including datasheets, reference manuals, videos, and application notes.

PSoC 1 Data Sheets

Visit the PSoC 1 Datasheets page for PSoC 1 datasheets that list the features and electrical specifications of all PSoC 1 device families.

Learning PSoC 1 Designer

Visit the PSoC Designer home page to download the latest version of PSoC Designer.

Launch PSoC Designer and navigate to the following items:

PSoC Designer User Guides: Visit the PSoC Designer User Guide page for a PSoC Designer User Designer.

IDE User Guide:

Launch PSoC Designer > Help > Documentation > Designer Specific Documents > IDE User Guide. This guide gives you the basics on developing PSoC Designer projects.

Example Projects:

Launch PSoC Designer > Design Catalog > Example projects. These example projects demonstrate how to configure and use PSoC Designer Components.

PSoC Designer Training

These trainings help to designs and demonstrate the Intro to PSoC Designer, Dynamic Reconfiguration and Debugging with PSoC.

Component Datasheets:

Launch PSoC Designer > Workspace Explorer. Under the Chip tab, right-click on a Component and select Datasheet.

Application Notes

Application notes assist you with understanding specific features of the device and designing your PSoC 1 application. Visit the PSoC 1 Application Notes page for a complete list of PSoC 1 application notes.

Knowledge Base Articles

These are a database of frequently asked technical support questions and their answers. Visit the Knowledge Based Articles page for a complete list of PSoC 1 knowledge based articles.

Technical Reference Manual (TRMs)

The TRM provides complete detailed descriptions of the internal architecture of the PSoC 1 devices. Visit the PSoC 1 Technical Reference Manuals page for a list of PSoC 1 TRMs.

Device Errata

These documents list any specification of the device that deviates from either the device datasheet or technical reference manual. Visit the Device Errata page for a list device errata documents.

Technical Support

If you have any queries or questions, our technical support team is happy to assist you. You can create a support request using the Cypress Technical Support link.

If you are in the United States, you can talk to our technical support team by calling our toll-free number +1-800-541-4736. Select option 8 at the prompt.

You can also use the following support resources if you need quick assistance:

- Self-help
- Local Sales Office Locations

My First PSoC 1 Design

This section shows you the step-by-step process of building a simple design with PSoC 1 using PSoC Designer, then programming it into a PSoC device, and configuring PSoC 1 demonstration boards to view the results of the application.

About the Design

This design creates a simple project that produces two PWM outputs (one using hardware resource and other using software), on two GPIOs. One GPIO will display a blinking LED at a fixed rate, while another GPIO will show a pulsing, or heartbeat, LED. The project also measures an analog voltage using an ADC user module.

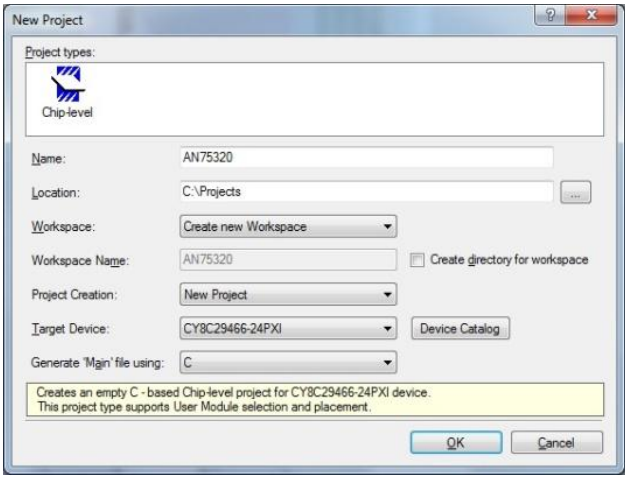
Creating My First PSoC 1 Design

Create Project

You start with creating a project in PSoC Designer. Here, you select the device, and then set the development language as C. This example project uses CY8C29466-24PXI, because that is the part that accompanies the CY3210-PSoCEval1 kit.

1. In PSoC Designer, select **File** > **New Project** and name it *AN75320*, as shown in Figure 3.

Figure 3. Creating a New PSoC Designer Project



Name: AN75320

Location: C:\Projects

Workspace: Create new Workspace

Workspace Name: AN75320 □ Create directory for workspace

Project Creation: New Project

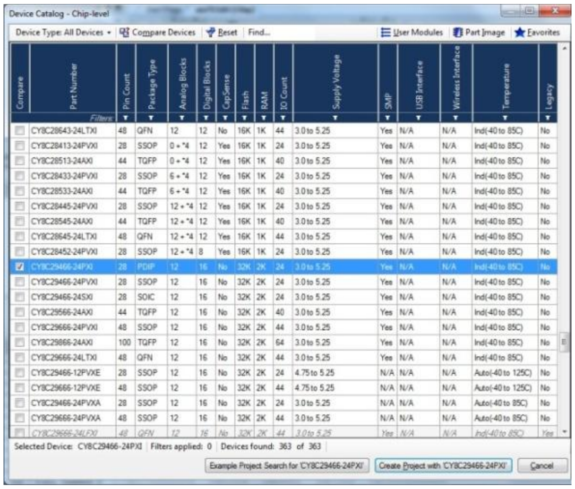
Target Device: CY8C29466-24PXI

Generate 'Main' file using: C

Creates an empty C - based Chip-level project for CY8C29466-24PXI device.
This project type supports User Module selection and placement.

2. Click on **Device Catalog** to select the device as shown in Figure 4.

Figure 4. Device Catalog



Compare	Part Number	Pin Count	Package Type	Analog Blocks	Digital Blocks	CapSense	Flash	RAM	IO Count	Supply Voltage	SMP	USB Interface	Wireless Interface	Temperature	Legacy
□	CY8C28643-24LTXI	48	QFN	12	12	No	16K	1K	44	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C28413-24PVXI	28	SSOP	0 + *4	12	Yes	16K	1K	24	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C28513-24AXI	44	TQFP	0 + *4	12	Yes	16K	1K	40	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C28433-24PVXI	28	SSOP	6 + *4	12	Yes	16K	1K	24	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C28533-24AXI	44	TQFP	6 + *4	12	Yes	16K	1K	40	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C28445-24PVXI	28	SSOP	12 + *4	12	Yes	16K	1K	24	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C28545-24AXI	44	TQFP	12 + *4	12	Yes	16K	1K	40	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C28645-24LTXI	48	QFN	12 + *4	12	Yes	16K	1K	44	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C28452-24PVXI	28	SSOP	12 + *4	8	Yes	16K	1K	24	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
☑	CY8C29466-24PXI	28	PDIP	12	16	No	32K	2K	24	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C29466-24PVXI	28	SSOP	12	16	No	32K	2K	24	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C29466-24SXI	28	SOIC	12	16	No	32K	2K	24	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C29566-24AXI	44	TQFP	12	16	No	32K	2K	40	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C29666-24PVXI	48	SSOP	12	16	No	32K	2K	44	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C29866-24AXI	100	TQFP	12	16	No	32K	2K	64	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C29666-24LTXI	48	QFN	12	16	No	32K	2K	44	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	No
□	CY8C29466-12PVXE	28	SSOP	12	16	No	32K	2K	24	4.75 to 5.25	N/A	N/A	N/A	Auto(-40 to 125C)	No
□	CY8C29666-12PVXE	48	SSOP	12	16	No	32K	2K	44	4.75 to 5.25	N/A	N/A	N/A	Auto(-40 to 125C)	No
□	CY8C29466-24PVXA	28	SSOP	12	16	No	32K	2K	24	3.0 to 5.25	N/A	N/A	N/A	Auto(-40 to 85C)	No
□	CY8C29666-24PVXA	48	SSOP	12	16	No	32K	2K	44	3.0 to 5.25	N/A	N/A	N/A	Auto(-40 to 85C)	No
□	CY8C29666-24LFXI	48	QFN	12	16	No	32K	2K	44	3.0 to 5.25	Yes	N/A	N/A	Ind(-40 to 85C)	Yes

Selected Device: CY8C29466-24PXI | Filters applied: 0 | Devices found: 363 of 363

Example Project Search for 'CY8C29466-24PXI'
Create Project with 'CY8C29466-24PXI'
Cancel

Note If you are developing this project with another kit, use the following part numbers:

- **CY3214-PSoCEvalUSB:**

CY8C24994-24LTXI

- **CY3271-PSoC First Touch:**

CY8C20634-12FDXI

3. Click on **Create Project with 'CY8C29466-24PXI'** and then click **OK**.

The project workspace opens, and displays the Chip Editor and the Workspace Explorer. Press Ctrl and click to zoom in the Chip Editor view, which shows the block diagram of the selected device.

Select User Module

In this step, you select the required components from the User Module catalog and configure them. This project uses the following user modules:

- One ADC user module
- Two PWM8 user module
- One LCD user module

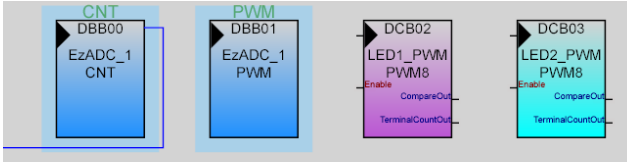
The user module window is in the right lower side; select **View** > **User Module Catalog** to display it.

1. On the User Module Catalog, expand the **ADCs** folder.
2. Right-click **EzADC** and select **Place**. The EzADC Configuration window opens.
3. In the Configuration window, select **Double Stage Modulator** and click **OK**. (This configuration uses two switched-capacitor blocks for better performance).
4. Under the **PWMs** folder, right-click **PWM8**, and select **Place**.

5. Repeat the placement process to place two **PWM8** User Modules in the design.
6. From the **Misc Digital** folder, right-click **LCD**, and select **Place**.

After you have placed the required user modules, the Designer window will look like Figure 5. In addition, the placed user modules are displayed in the Workspace Explorer. Note that each user module placed is named as ComponentName_1, and so on. For example, the EzADC user module is listed as EzADC_1.

Figure 5. EzADC and PWM User Module Placement in the Chip Editor



Set User Module Parameters

In this step, you configure the parameters for the user modules that you placed in the design.

When you click on a user module from either the Workspace Explorer or the Chip Editor, the Parameters window for the selected user module appears on the left hand side, where you can edit the parameters.

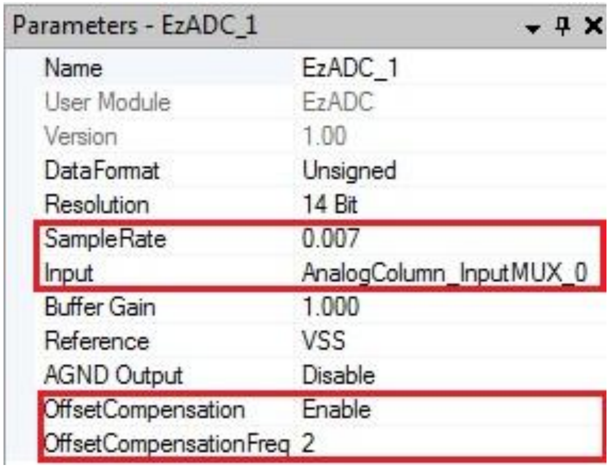
Refer to the diagrams for the parameters to set or change.

1. Select **EzADC_1** and change its parameters as per Table 3. After you have made the changes, the window looks similar to Figure 6. Parameters that are to be changed from their default values are highlighted.

Table 3. EzADC_1 User Module Parameters

Parameter	Value	Description
Name	EzADC_1	Give the User Module name
Data Format	Unsigned	Sets the output to unsigned data as input is positive only
Resolution	14 Bit	Sets the max resolution for EzADC
Sample rate	0.007	Sample rate depend on resolution. See the user module datasheet of EzADC.
Input	AnalogColumn_InputMUX_0	Select the analog input data bus to receive the input analog voltage for the source.
Offset Compensation	Enable	Enables the correlated double sampling
Buffer Gain	1.000	Sets the PGA gain to 1 for unity amplification
Reference	VSS	Sets the reference for the PGA to perform offset compensation
AGND Output	Disable	Disables the AGND output. This option can be used to bring out the internal AGND to connect bipolar signals to the ADC
Offset Compensation Frequency	2	Sets the number of ADC samples after which offset is corrected

Figure 6. EzADC_1 User Module Parameters



Parameters - EzADC_1

Name	EzADC_1
User Module	EzADC
Version	1.00
DataFormat	Unsigned
Resolution	14 Bit
SampleRate	0.007
Input	AnalogColumn_InputMUX_0
Buffer Gain	1.000
Reference	VSS
AGND Output	Disable
OffsetCompensation	Enable
OffsetCompensationFreq	2

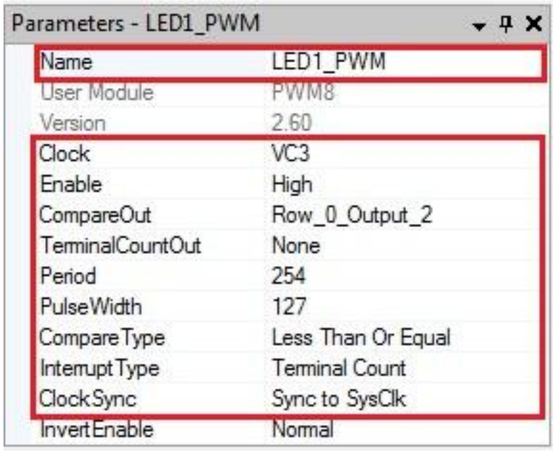
2. Select **PWM8_1** and change its parameters as per Table 4. After you have made the changes, the window appears as shown in Figure 7. Note that the user module is now renamed as **LED1_PWM**.

Table 4. LED1_PWM User Module Parameters

Parameter	Value	Description
Name	LED1_PWM	Assign a name to the User Module.
Clock	VC3	Choose the clock source as **VC3**.
Enable	High	Select enable as High for continuous count.
Compare Out	Row_0_Output_2	Connect the compare output to a GPIO via a row output bus.
Terminal Count Out	None	This is an auxiliary counter. You don't need to connect it to any of the row output buses.
Period	254	Set the period of the counter.

Parameter	Value	Description
Pulse Width	127	Sets the pulse width of the PWM output (50% duty cycle).
Compare Type	Less Than Or Equal	Select full range for the compare type function.
Interrupt Type	Terminal Count	This selection trigger the interrupt on the terminal count of the counter register.
Clock Sync	Sync to SysClK	Synchronizes the clock input to system clock
Invert Enable	Normal	This determines the sense of the enable input signal.

Figure 7. PWM_1 User Module Parameters

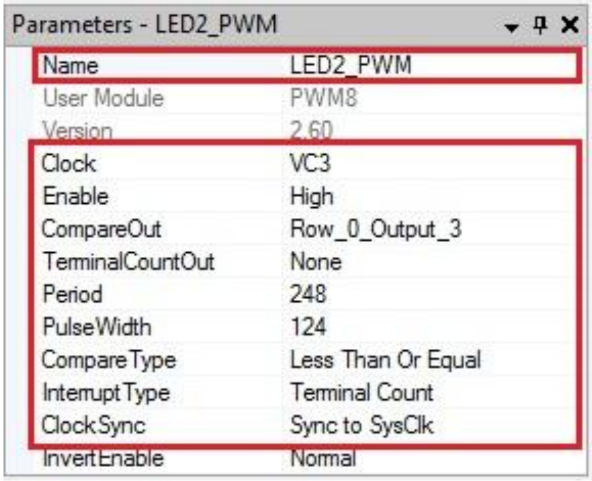


Parameters - LED1_PWM

Name	LED1_PWM
User Module	PWM8
Version	2.60
Clock	VC3
Enable	High
CompareOut	Row_0_Output_2
TerminalCountOut	None
Period	254
PulseWidth	127
CompareType	Less Than Or Equal
InterruptType	Terminal Count
ClockSync	Sync to SysClk
InvertEnable	Normal

3. Select **PWM8_2** and edit its parameters to the settings as per Figure 8 shows. Note that the user module is now renamed to **LED2_PWM**.

Figure 8. PWM_2 User Module Parameters



Parameters - LED2_PWM

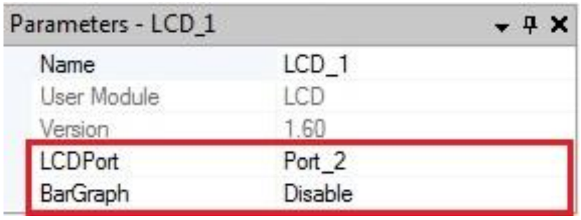
Name	LED2_PWM
User Module	PWM8
Version	2.60
Clock	VC3
Enable	High
CompareOut	Row_0_Output_3
TerminalCountOut	None
Period	248
PulseWidth	124
CompareType	Less Than Or Equal
InterruptType	Terminal Count
ClockSync	Sync to SysClk
InvertEnable	Normal

4. Select the **LCD**_1 user module and change its parameters as per Table 5. After you have made the changes, the window looks similar to Figure 9.

Table 5. LCD_1 User Module Parameter

Parameter	Value	Description
Name	LCD_1	Give the User Module name
LCD Port	Port_2	This sets interfaces the LCD display module to assigned Port
BarGraph	Disable	Disables the bar graph function

Figure 9. LCD Parameters

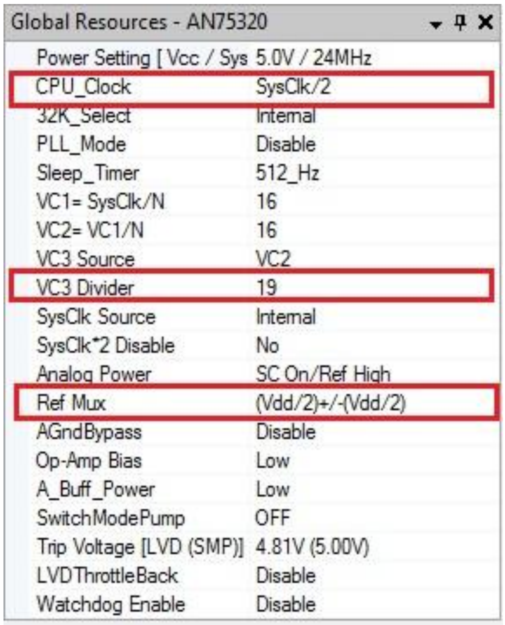


Parameters - LCD_1

Name	LCD_1
User Module	LCD
Version	1.60
LCDPort	Port_2
BarGraph	Disable

5. Open Global Resources window from **View > Global Resources** to configure the source clock VC3 for PWMs clock frequency and the Ref Mux for ADC input data range. Leave other parameter to their default value. After changes, the Global Resources window looks similar to Figure 10. For more information on Global Resources, see the IDE User Guide.

Figure 10. Global Resources Parameters



Global Resources - AN75320

Power Setting [Vcc / Sys	5.0V / 24MHz
CPU_Clock	SysClk/2
32K_Select	Internal
PLL_Mode	Disable
Sleep_Timer	512_Hz
VC1= SysClk/N	16
VC2= VC1/N	16
VC3 Source	VC2
VC3 Divider	19
SysClk Source	Internal
SysClk*2 Disable	No
Analog Power	SC On/Ref High
Ref Mux	(Vdd/2)+/-(Vdd/2)
AGndBypass	Disable
Op-Amp Bias	Low
A_Buff_Power	Low
SwitchModePump	OFF
Trip Voltage [LVD (SMP)]	4.81V (5.00V)
LVDThrottleBack	Disable
Watchdog Enable	Disable

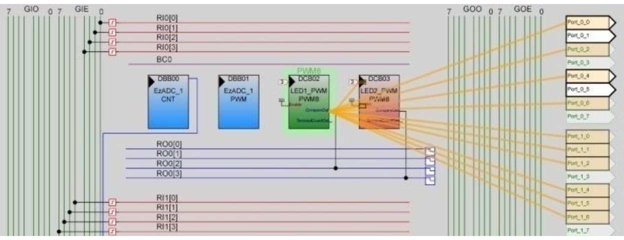
Route PWM Signals

In this section, you route the PWM signals from PWM user modules to selected GPIO. To see the signal on a GPIO pin, you must implement routing for the PWM signals.

In addition, this design implements the logic in the row outputs by feeding the two PWM outputs into an XOR. The difference of the period of the two PWM signals, XORed together, will create the effect of pulsing (heartbeat).

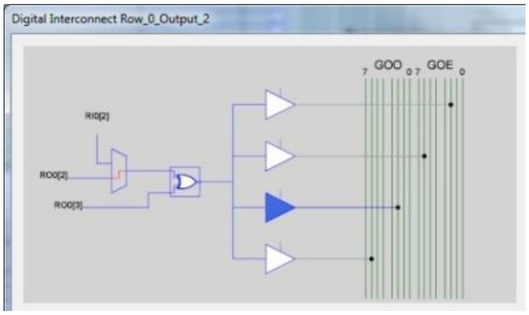
1. Connect CompareOut of LED1_PWM to Row_0_Output_2.
2. Connect the CompareOut of PWM8_2 to Row_0_Output_3.
3. Connect the CompareOut of LED1_PWM to Port_1_2 using auto routing. Press **Shift** and select the CompareOut pin of LED1_PWM and connect it with Port_1_2 as Figure 11 shows.

Figure 11. PWMs Connected to Row Outputs



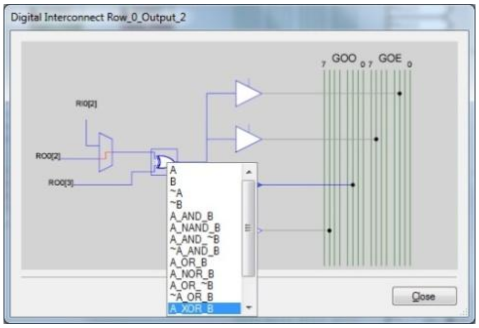
4. Click on the Row_0_Output_2 Digital Interconnect to open the configuration options as shown in Figure 12.

Figure 12. RO0[2] Digital Interconnect View



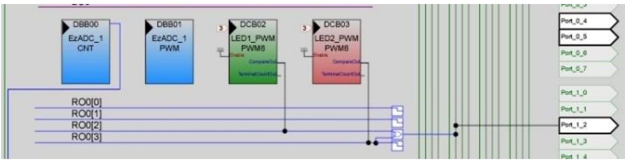
5. Click on the LogicTable_Select (indicated by the square box in the middle) and select A_XOR_B from the drop-down list as shown in Figure 13, and then click **Close**.

Figure 13. Digital Interconnect Configuration



6. Confirm that the digital routing diagram looks as Figure 14 shows. If they match, then the configuration of the pulsing heartbeat LED is now complete.

Figure 14. Designer Chip View



Add Coding

In this step, you add the code to configure all the user modules and GPIOs to implement a software LED that will produce a blinking LED. You add the custom code that follows on to the *main.c* file for the project.

1. In Workspace Explorer, locate the **Source Files** folder under the **AN75320 Folder** and open the *main.c* file.
2. Copy the C code listed below and pastes it into the *main.c* file, replacing the existing code.

Code 1. Project 1 main.c

```
/* part specific constants and macros */
#include <m8c.h>

/* PSoC API definitions for all User Modules */
#include "PSoCAPI.h"

/* Definitions for all input and output operation */
#include "stdio.h"

/* Macros to set ADC parameters */
```

```
#define GAIN                  1
#define MAX_ADC_COUNTS  16383
#define ADC_RANGE       5000

/* Macros to select port 1 */
#define PORT_1_3        PRT1DR

/* Variable to store the ADC result */
WORD adc_result;

/* Variable to store the measured input in millivolts */
WORD v_in;
void main(void)
{
    static unsigned int index;

      /* Buffer used for the long to ASCII conversion */
      char LCDBuffer[17];

      /* Initializes LCD to use the multi-line 4-bit interface  */
      LCD_1_Start();

      /* Enable global interrupts */
      M8C_EnableGInt ;

      /* Set the position to print the character */
      LCD_1_Position(0,0);

      /* print the Hello World in the first line */
      LCD_1_PrCString("Hello World!");

      /* Starts the LED1_PWM and LED2_PWM, high enable input begins the Counter */
      LED1_PWM_Start();
      LED2_PWM_Start();

      /* Initializes and starts the EzADC User Module resources */
      EzADC_1_Start(EzADC_1_HIGHPOWER);

      /* Starts the ADC conversion */
      EzADC_1_GetSamples(0);
                  while(1)
              {
                        /* Wait for the ADC result to be available */
                        while(!(EzADC_1_fIsDataAvailable()));

                        /* Read the ADC result and clear the data ready flag */
                        adc_result = EzADC_1_iGetDataClearFlag();

                        /* Calculate input voltage in mV */
                        v_in = (DWORD)adc_result*ADC_RANGE / MAX_ADC_COUNTS / GAIN;

                        /* Convert the input voltage to an ascii string */
                        csprintf(LCDBuffer, "ADC INPUT:%4dmV", v_in);
              /* Set the position to print the character */
              LCD_1_Position(1,0);

              /* print the voltage in the second line */
              LCD_1_PrString(LCDBuffer);

              /* Toggle the pin 1[3]*/
              PORT_1_3 ^= 0x08;

              /* Give some delay to view toggling effect*/
          for(index = 0; index < 22000; index++);
          }
}
```

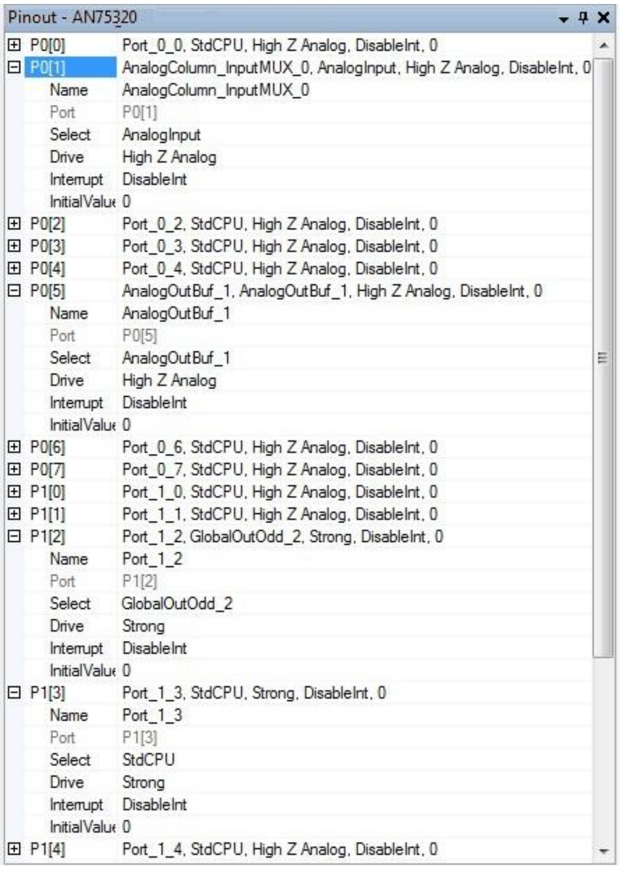
3. Include *local.mk* file attached with the project to enable floating point for the ADC. See the *local.mk* file for more information on this function.

Configure Pinout

In this step, you configure the GPIO drive mode for the pin to drive the LED. You configure the pinout for the selected device from the Pinout view. (**View** > **Pinout**). The Pinout panel appears on the lower-left side of the workspace.

1. On the Pinout view, expand the options for P0[1], P0[5,]P1[2], and P1[3]. Configure the pins as shown in Figure 15.

Figure 15. Pin Configuration



Pinout - AN75320

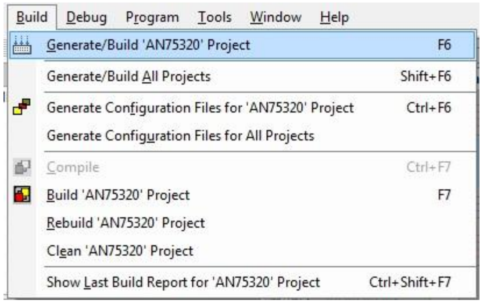
⊞ P0[0]	Port_0_0, StdCPU, High Z Analog, DisableInt, 0
⊟ P0[1]	AnalogColumn_InputMUX_0, AnalogInput, High Z Analog, DisableInt, 0
Name	AnalogColumn_InputMUX_0
Port	P0[1]
Select	AnalogInput
Drive	High Z Analog
Interrupt	DisableInt
InitialValue	0
⊞ P0[2]	Port_0_2, StdCPU, High Z Analog, DisableInt, 0
⊞ P0[3]	Port_0_3, StdCPU, High Z Analog, DisableInt, 0
⊞ P0[4]	Port_0_4, StdCPU, High Z Analog, DisableInt, 0
⊟ P0[5]	AnalogOutBuf_1, AnalogOutBuf_1, High Z Analog, DisableInt, 0
Name	AnalogOutBuf_1
Port	P0[5]
Select	AnalogOutBuf_1
Drive	High Z Analog
Interrupt	DisableInt
InitialValue	0
⊞ P0[6]	Port_0_6, StdCPU, High Z Analog, DisableInt, 0
⊞ P0[7]	Port_0_7, StdCPU, High Z Analog, DisableInt, 0
⊞ P1[0]	Port_1_0, StdCPU, High Z Analog, DisableInt, 0
⊞ P1[1]	Port_1_1, StdCPU, High Z Analog, DisableInt, 0
⊟ P1[2]	Port_1_2, GlobalOutOdd_2, Strong, DisableInt, 0
Name	Port_1_2
Port	P1[2]
Select	GlobalOutOdd_2
Drive	Strong
Interrupt	DisableInt
InitialValue	0
⊟ P1[3]	Port_1_3, StdCPU, Strong, DisableInt, 0
Name	Port_1_3
Port	P1[3]
Select	StdCPU
Drive	Strong
Interrupt	DisableInt
InitialValue	0
⊞ P1[4]	Port_1_4, StdCPU, High Z Analog, DisableInt, 0

Build and Program

In this step, you connect the MiniProg1 device to your computer, and the program PSoC device with the built project.

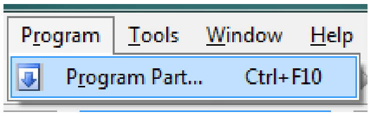
1. Select **Build** > **Generate/Build 'AN75320' Project** or press **F6**. (See Figure 16).

Figure 16. Build and Generate Option


Build
Debug
Program
Tools
Window
Help
Generate/Build 'AN75320' Project F6
Generate/Build All Projects Shift+F6
Generate Configuration Files for 'AN75320' Project Ctrl+F6
Generate Configuration Files for All Projects
Compile Ctrl+F7
Build 'AN75320' Project F7
Rebuild 'AN75320' Project
Clean 'AN75320' Project
Show Last Build Report for 'AN75320' Project Ctrl+Shift+F7

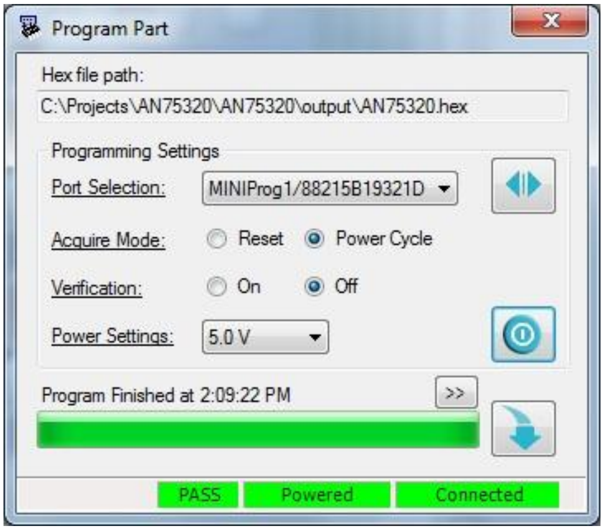
2. Connect MiniProg1 to your computer.
3. Select **Program** > **Program Part** (See Figure 17). The Program Part window appears as shown in Figure 18.

Figure 17. Program Part Option



4. On the Program Part window, do the following:
   a. Click the **Connect** button (next to the Port Selection field) to connect the device.
   b. Set the **Acquire Mode** to *Power Cycle* if the MiniProg is supplying power to the device and it can acquire the device by cycling power. This is the default option. For this project, use the default option.
   c. Set the **Acquire Mode** to *Reset* if the device is externally powered and the MiniProg can only acquire the device by resetting it.
5. Click the **Program** button to program the device as shown in Figure 18.

Figure 18. Programming Status


Program Part

Hex file path:
C:\Projects\AN75320\AN75320\output\AN75320.hex

Programming Settings

Port Selection: MINIProg1/88215B19321D

Acquire Mode: ○ Reset ● Power Cycle

Verification: ○ On ● Off

Power Settings: 5.0 V

Program Finished at 2:09:22 PM

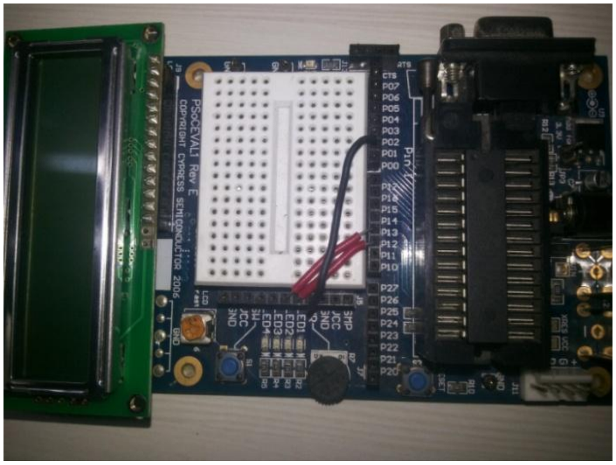
PASS | Powered | Connected

6. When the program operation is completed, press the **Power** button to toggle power for the device.

Setting Up the CY3210-PSoCEval1 Board

This demonstration is compatible with the CY3210-PSoCEval1 hardware. For more information on this kit, see http://www.cypress.com/?rID=2541.

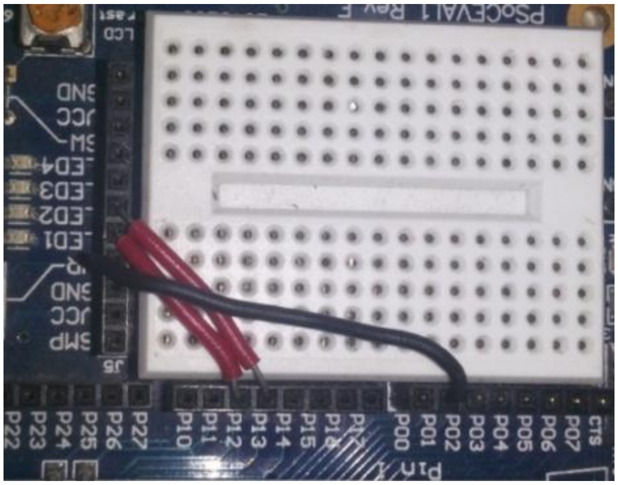
Figure 19. A View of the CY3210-PSoCEval1 Kit



Perform the following steps to configure and program the CY3210-PSoCEval1 kit. Note that this kit requires a programming device, such as a MiniProg or ICE-Cube.

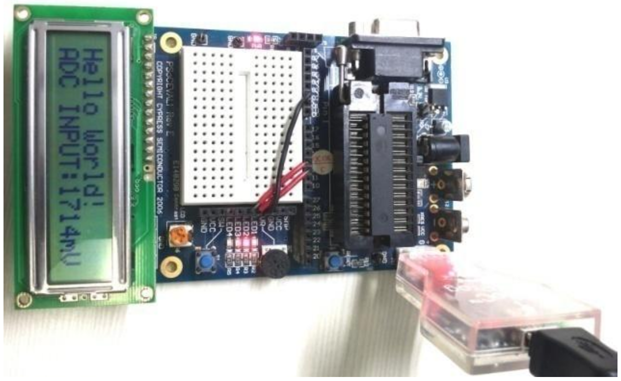
1. With no cables connected to the demo board, ensure that J1, J2, and J3 are not in position.
2. Place a wire connecting P0[1] to VR, P1[2] to LED1 and another wire connecting P1[3] to LED2 (see Figure 20).

Figure 20. CY3210-PSoCEval1 Pin Connections



3. Ensure that a CY8C29466-24PXI is the device currently on the board.
4. Connect a 12-V DC power supply to J10 on the kit. Alternatively, you can connect a 9-V battery to J12. In addition, connect a programming device to J11. The board connections can be seen in Figure 21.

Figure 21. Power and Program Connections

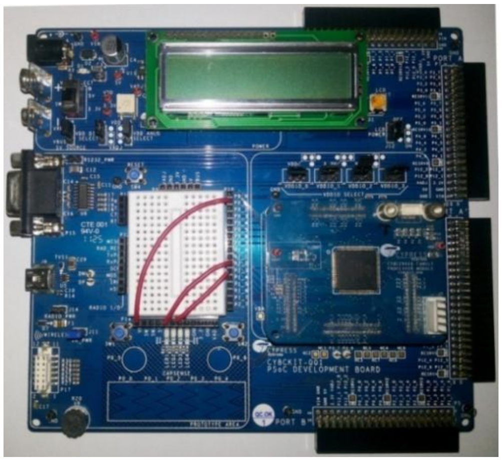


Setting Up the CY8CKIT-001 Board

When you use a CY8C29x66 processor module, this demonstration is compatible with the CY8CKIT-001 hardware. For more information on this kit, see http://www.cypress.com/?rID=37464.

Perform the following steps to configure and program the CY8CKIT-001 kit. Note that this kit requires a programming device, such as a MiniProg or ICE-Cube. Additionally, this kit supports on-chip debug when using the ICE-Cube.

Figure 22. A View of the CY8CKIT-001 Kit



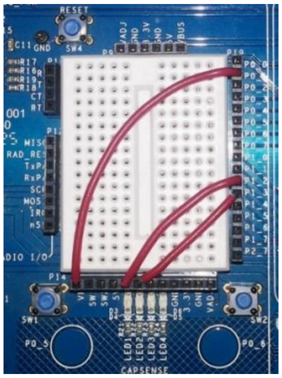
1. With no cables connected to the demo board, set the following jumpers to configure the development board as shown in Table 6:

Table 6. Jumper Settings

Jumper	Setting
VDD Select (SW3)	5-V (Up Position)
5-V Source (J8)	VREG (Upper two pins
VDD Digital (J7)	VDD (Upper two pins)
VDD Analog (J6)	VDD (Upper two pins)
LCD Power (J12)	ON (Lower two pins)
VDDIO Select (J2-J5)	VDD (Upper-left two pins)

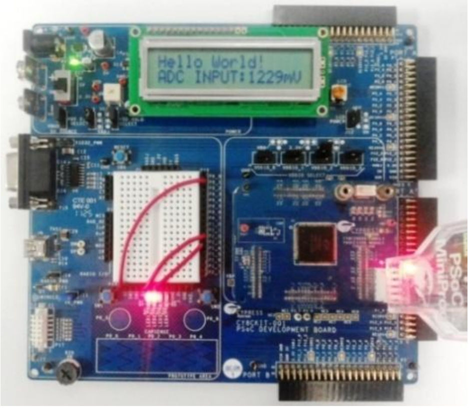
2. Place a wire connecting P0[1] to VR, P1[2] to LED1 and another wire connecting P1[3] to LED2 (See Figure 23).

Figure 23. Alternative View of Pin Connections



3. Connect a programming device, such as a MiniProg or ICE, to J5 on the PSoC 1 processor module.
4. Connect power to the board with either a 12-V DC power supply or a 9-V battery or using power from programmer. See Figure 24 for board connections.

Figure 24. Power and Program Connections



Regardless of which development kit you use to test the application, you will observe the following:

1. The LCD displays “Hello World” and “ADC INPUT:----mV”
2. One LED blinks rapidly.
3. The second LED pulses with the light intensity alternatively and slowly increasing and decreasing.

Summary

This application note has explored the PSoC 1 architecture and development tools. This also described the tools you need to start PSoC 1 projects. PSoC 1 is a true programmable embedded system-on-chip integrating configurable analog and digital peripheral functions, memory, and an M8C microcontroller on a single chip.

This application note has also guided you to a comprehensive collection of resources to accelerate in-depth learning about PSoC 1.

For more information on any of the topics mentioned, refer to any of the links or references made, or visit www.cypress.com/go/psoc1.

Related Application notes

This application note covers the basic details about PSoC 1. You can refer to specific Application Notes for more information and projects regarding to a specific topic.

1. Getting Started

AN54181 - Getting Started with PSoC 3

AN79953 – Getting Started with PSoC 4

AN77759 - Getting Started with PSoC 5LP

2. Analog

AN2219 - PSoC® 1 Selecting Analog Ground and Reference

AN74170 – PSoC 1 Analog Structure and Configuration with PSoC Designer™

AN13666 - PSoC® 1 Driving Analog Buffer Output to the Rail

AN2096 - PSoC® 1 - Using the ADCINC Analog to Digital Converter

3. Switched Capacitor Blocks

AN2041 – Understanding PSoC 1 Switch Capacitor Analog Blocks

AN2168 – PSoC 1 Understanding Switched Capacitor Filters

AN16833 - Signal Mixing with PSoC® Switched Capacitor Blocks

4. GPIO

AN2094 - PSoC® 1 - Getting Started with GPIO

5. Programming

AN44168 - PSoC® 1 Device Programming using External Microcontroller (HSSP).

6. Digital

AN2141 - PSoC® 1 Glitch Free PWM

7. Flash

AN2015 - PSoC 1 Reading and Writing Flash & E2PROM

8. I2C

AN50987 – Getting Started with I2C in PSoC 1

9. SPI

AN51234 - Getting Started with SPI in PSoC® 1

10. Sleep Mode

AN47310 - PSoC® 1 Power Savings Using Sleep Mode

11. LCD

AN56384 - PSoC® 1 Segment LCD Direct Drive

AN2152 - PSoC® 1 Graphics LCD and PSoC® Interface

About the Author

Name:	Robert Murphy
Title:	Application Engineer Sr.
Background:	Robert Murphy graduated from Purdue University with a Bachelor’s Degree in Electrical Engineering Technology.
Contact:	rlrm@cypress.com

Document History

Document Title: Getting Started With PSoC® 1 – AN75320

Document Number: 001-75320

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	3498585	RLRM	1/17/2012	New application note
*A	3846596	GULA	12/19/2012	Updated project to PSoC Designer 5.3. Updated all the relevant screenshots. Removed incorrect mention of PSoC Creator Added references to all Application Notes and Technical Reference Manual. Corrected the SC-based Integrator circuit. Minor content edits.
*B	4064460	GULA	07/16/2013	Updated the UM parameters screenshots Updated the board images Added a section on Auto-routing Added links for the App Notes and kits Replaced section on CY3215-DK with details about CY3215A-DK. Added Next Steps section for details about the specific App Notes
*C	4365121	ASRI	04/29/2014	Updated all the section of the AN Added ADC functionally into AN project and modified the project Added links for the various document for PSoC 1 learning resources and kits

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Automotive	cypress.com/go/automotive
Clocks & Buffers	cypress.com/go/clocks
Interface	cypress.com/go/interface
Lighting & Power Control	cypress.com/go/powerpsoc cypress.com/go/plc
Memory	cypress.com/go/memory
PSoC	cypress.com/go/psoc
Touch Sensing	cypress.com/go/touch
USB Controllers	cypress.com/go/usb
Wireless/RF	cypress.com/go/wireless

PSoC® Solutions

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/go/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.