PERSISTOR
Instruments Inc.

254-J Shore Road, Bourne, MA 02532, USA
Tel: 508-759-6434 Fax: 508-759-6436
www.persistor.com info@persistor.com

Data Acquisition and Storage Solutions for Industry and Science

CF2 Version 2.27r1 Release Notes

April 29, 2002

Introducing the CF2

We've gone to great lengths to make the CF2 an almost drop in replacement for the CF1 and except for the different timer modules, you should find that all of the BIOS and PicoDOS support software work the same on both systems. Even though the CF2 is a brand new machine, it inherits most of its hardware design and firmware code from the mature CF1. They are in fact built from the same code tree in the same CodeWarrior project. This document exists primarily to point out the differences.

If you're already a CF1 user, you should have no problem migrating to the CF2 platform. However, if you will be developing projects concurrently for both CF1 and CF2, we suggest adopting multi-target projects to keep your code organized and ensure that CodeWarrior builds correctly as you move from one to the other.

If you're a first time customer of Persistor Instruments, you may find some of the current CF2 documentation a little perplexing. The CF2 is very similar to its CF1 predecessor and we are presently relying on the original CF1 User's Manual to fill the void while we rewrite the documentation specifically for the CF2. New manuals will be posted on our web site as they're completed.

Getting Started with the CF2

1. **Read these CF2 release notes**. They will alert you to any important changes that may have not made it into the current documentation.

2. **Read the CF2 Getting Started Guide**. This will help you install the development software and get you acquainted with the key capabilities of the CF2.

3. **Read the html CF1 User's Manual**. After installing the PicoDEV software, you'll find a link to this and all of the other CF2 documentation in your start menu at: Programs->Persistor->CFxDocumentationIndex.htm.

   a. The sections of the html manual entitled; "Features & Specifications", "Hardware Reference" and "Getting Started with the CF1" should be ignored. Instead, pay more attention to the CF2 Getting Started Guide, which is accurate on those subjects.

   b. The sections on; "PicoDOS Commands" and "Development" are mostly appropriate for the CF2, except where CTM functions are discussed.

   c. The example programs are functional on the CF2, except once again, where the CTM is referenced. Each of these example programs were used as functional tests on the CF2 to insure its functionality matched the older CF1.

4. **Explore some of the examples**. You will find these projects in your Persistor directory. If you followed the default installations, they will be at C:\Program Files\Persistor\MotoCross Support\CFX\Examples.

   a. Examples of TPU code are located here, and the comments in the examples are the only documentation to date.

   b. If you are using the R216AU RecipeCard with ADS8344 16 bit A/D converter, you will want to read ADExamplesReadMe.txt to learn how to switch the examples to the correct converter. Similarly, PicoDAQ has been rewritten to allow A/D selection in the header file.

5. **Check www.persistor.com for updates**. We'll periodically post updated documentation and new programming examples.

We believe the CF2 is one of the best values available for embedded computer development. Our goal is to give you the tools and support you need to quickly get your projects completed. Please do not hesitate to contact us for technical help, advice, or just some clarification on the documentation. You can e-mail us at help@persistor.com or telephone tech support us at 413-637-9171 or 508-759-6434 between 9 and 5 on standard business days.

Major CF1 versus CF2 differences

Overview

The two major differences between the CF1 and CF2 relate to the different real-time clocks and the different timer modules. The remaining minor differences relate to the hardware implementation issues necessary to provide near pin-for-pin plug-in compatibility.

Pin C5 is now VBAK

This was an unused pass-through pin on the CF1 but it now brings out VBAK, which is the output from the MAX6367 supervisor that powers the SRAM and MSP430 backup circuitry when the main supply goes away.

Timer Modules

The CF1's 68338 featured a Counter-Timer Module (CTM) that performed basic digital I/O, variable frequency pulse width modulation, and pulse width or period measurement with 15 signals brought to connector C. The CF2's 68332 features a Time Processor Unit (TPU) that can perform all of the CTM functions as well as a variety of other pre-programmed and custom time related functions.

… and the UARTs for free

Perhaps the most impressive and useful feature is the ability of each channel to function as an independent UART receive or transmit channel and providing the CF2 with up to 8 full duplex UARTs. There's nothing for documentation yet, but this version of PicoDOS contains high level C drivers for opening, closing, and managing buffered transfers. The API is defined near the bottom of <cfxpico.h> and there's an example project in Examples\TPU directory demonstrating three levels of complexity. Expect more examples and full documentation soon.

Pin C36 (was CTM31L)

Missing from the CF2 is an equivalent connection to pin C36 which had been the CTM31L input signal. This connection was the load input to the 68338's Counter Timer Module (CTM). It is now a "DON'T CONNECT" for the CF2. There exists a manufacturing provision on the CF2 boards to connect this pin to the TPU's T2CLK input at the expense of loosing the very clean 40kHz LFCLK that now goes to T2CLK and feeds the custom local-RTC TPU function used by PicoDOS.

... speaking of custom TPU functions

Also in the Examples\TPU directory are CodeWarrior projects containing the microcode source to all of the Motorola TPU functions. CodeWarrior can't assemble these files, but we've included Motorola's freeware TPU assembler for users who feel the need to roll their own suite of TPU functions. All of the Motorola PDF documentation for the TPU functions can be found in the CFX\Docs\pdf\TPU directory. Unfortunately, the current BIOS release has no mechanism to install custom code, but it's high on the priority list and we'll make this fairly easy to do.

Real-time Clocks

The CF1's 68338 featured a built-in RTC module that accepted a separate Vdd and 32.768kHz crystal allowing the clock to run when main power was removed from the processor. The CF1's 68338 CPU ran from a different 40kHz crystal and relied on a companion PIC microcontroller for various simple supervisory support.

The CF2's 68332 has no such integral RTC counterpart so we beefed up the capability of the separate supervisory processor with a TI MSP430F123 and gave it the job of maintaining time. This had the immediate benefit of unified clock source for all of the CF2's circuitry but at the cost of access time through a serial interface versus the direct interbus access with the 68338. To minimize the impact, we used the hardware SPI modules integral to both processors for 4MHz bandwidth, but we used a non-SPI control line from the 68332 to trigger the SPI chip select on the MSP430 to keep all of the four QSPI peripheral chip selects available for end user expansion.

QSPI considerations

The major implication of that change is that the CF2 cannot operate as an SPI slave, that QPB slot 15 has been commandeered for onboard use, and that three of the SPI control pins (SCK, MOSI, and MISO) are no longer available as general purpose I/O pins, even if your CF2 system hooks to no external SPI devices. Since QPB slot 15 is the idle state for normal QSPI operations in both multiplexed and non-multiplexed SPI systems, this should not have any major impact.

Access to time considerations

The CF1's 68338 had direct intermodule access to time information which took only several tens of microseconds to fetch. Because of that, we built some very convenient countdown, elapsed, and delay timers into the CF1 BIOS which could operate with extremely short intervals. Since SPI serial linkage to time information drives that up into the milliseconds range we built a custom TPU function that runs in parallel with the SPI RTC and which is inherently synchronized by virtue or running from the exact same clock signal. It only needs to refresh the local time with an SPI transfer after the CF2 comes out of reset or low power modes where the master or TPU clocks are stopped.

RAM Size and Operating Temperature

The original CF1 design was populated with componentry that allowed us to hit a specific pricing target. This precluded using the then new and expensive 512KB SRAMs and the more costly and then longer lead time industrial temperature parts. Times change, prices and lead times have dropped, and we've been able to cut our manufacturing costs to the point where we can offer a 512KB, -40C to +85C CF2 at the same price as the old standard CF1. We're not expecting too many complaints about that change.

Give and Take

When we doubled up on the physical ram size, we doubled up on the ram taken by PicoDOS for system use. On the CF1, the first 64KB was reserved for PicoDOS leaving 192KB for your application. On the CF2, we reserve both the top and bottom 64KB, so your application now gets 384KB of ram.

BIOS dependency on PicoDOS

PicoDOS has always been dependent on the presence in flash of a viable BIOS image, but the converse had not previously been a requirement. It now is for the CF2. This is necessary to allow BIOS access to QPB, which is the SPI communications software that will allow the onboard SPI RTC to peacefully coexist with your SPI peripherals. QPB could arguably be moved into the BIOS, but it physically doesn't fit and it requires access to dynamic memory that BIOS can't get hold of. We're not expecting to really impact anybody but let us know if that's not really the case.

Known Bugs and Limitations in 2.27r1

Suspend Mode TPU Limitations

In suspend mode, you can make the 68332 request that the MSP430 turn off the LT1521 linear regulator which removes power to everything except the MSP430 and the RAM so long as you have a backup battery or external 3V at VBBK. You also tell it what conditions are to end the suspend mode, usually an alarm time, and low signal on wake, or a CompactFlash card insertion or removal. That's the easy part, and that we can actually already do. The hard part is capturing the entire state of the processor and setting up a mechanism to restore it on completion so that it looks like nothing ever happened. We could do that with the CF1's 68338, but the ever more capable TPU is also ever more challenging to wrestle into submission.

This means that you will need to take responsibility for reinitializing TPU channel functions on completion of a suspend operation. The suspend command will take of the lowest level TPU initialization and revert all channels back to simple digital inputs. It also takes care of freeing up any allocated TPU UART channels and their associated memory. You should consider grouping your TPU initialization code around a single void TPUSETUP(void) function that can then be setup as an automatic power post-change chore using PWRPostChgAddChore().

Developing with both CF1 and CF2

Most of the early CF2 customers are migrating from the CF1 and many will have active projects in development or maintenance phases that will require fairly frequent context switching. The latest release of the PicoDEV tools was designed to deal with exactly that. All of the examples and stationery projects included on the CD are CF1/CF2 neutral – they work with either. We use the Source Tree feature in CodeWarrior to setup search path preferences at both a global (IDE) and local (project) level. The examples and stationery do not have any predefined local source trees specified, so everything works from the global level.

Ahead are the instructions for changing the global level between CF1 and CF2 from the IDE Edit:Preferences menu. We don't include the directions for doing this at the project level, but it's basically the same operation except from a Project Preferences panel.

Switching between CF1 and CF2

IDE Switch

Edit:Preferences…
In IDE Preference Panels: click General:Source Trees
Select MxTarget, click Choose… button
Select (CF1) or (CF2) and click OK button
Click Change button (expect warnings)
Click OK button