This application report is intended to provide a deeper look into the components that differentiate the C2000 Microcontroller Unit (MCU) as it pertains to Real-Time Control Systems. The focus is on features that Texas Instruments (TI) believes have measurable impact to overall system performance as well as increased bandwidth to the MCU. This document also expands on the specific aspects of the system that are improved with the covered topics. Finally, appropriate references and links to discrete Part Numbers (PNs), reference designs, and demonstration kits for each topic are provided.

Contents
1 C2000™ MCUs and Real-Time Control ................................................................. 3
   1.1 Processing ........................................................................................................ 4
   1.2 Actuation ...................................................................................................... 4
   1.3 Sensing ....................................................................................................... 5
   1.4 Communications ....................................................................................... 5
2 Processing Key Technologies ............................................................................... 6
   2.1 Accelerated Trigonometric Math Functions ................................................. 6
   2.2 Improving Control Performance With Nonlinear PID Control ................. 8
   2.3 Understanding Flash Memory Performance In Real-Time Control Applications .................................................................................. 10
   2.4 Flexible System Interconnect .................................................................... 12
   2.5 Increasing Control Loop Bandwidth With An Independent Processing Unit .................................................................................. 14
   2.6 Deterministic Program Execution ............................................................. 16
3 Actuation Key Technologies ................................................................................ 18
   3.1 On-Chip Hardware Customization ............................................................ 18
   3.2 Reducing Limit Cycling in Control Systems .............................................. 20
   3.3 Shoot Through Prevention for Current Control Topologies ....................... 21
4 Sensing Key Technologies .................................................................................. 24
   4.1 Realizing Rotary Sensing Solutions ............................................................ 24
   4.2 Optimizing Acquisition Time vs Circuit Complexity for Analog Inputs .... 26
   4.3 Accurate Digital Domain Representation of Analog Signals .................... 28
   4.4 Resolving Tolerance and Aging Effects During ADC Sampling ............... 30
   4.5 Fast Detection of Over and Under Currents and Voltages ....................... 32
   4.6 Smart Sensing Across An Isolation Boundary ........................................... 33
5 Communications .................................................................................................. 35
   5.1 Distributed Real-Time Control Across An Isolation Boundary .................. 35
6 References ......................................................................................................... 37
   6.1 Device List .................................................................................................. 37
   6.2 Hardware/Software Resources .................................................................. 37
   6.3 Documentation ........................................................................................... 37

List of Figures
1 Simple Real-Time Control System ..................................................................... 3
2 C2000 MCU Components .................................................................................. 4
3 Park Transform .................................................................................................. 6
4 TMU improvement for Park Transform ............................................................. 6
5 Non-Linear PID Block Diagram ................................................................. 8
6 Comparison of Response Time Between Linear and Non-linear PID ................. 9
7 C2000 Flash Prefetch Module ................................................................. 10
8 Input X-Bar on the TMS320F2837xD MCU ............................................... 12
9 Local Mux and Logical OR on the TMS320F2837xD MCU .......................... 13
10 X-Bar Sources and Destinations on the TMS320F2837xD MCU .................. 13
11 C28x and CLA Interfacing With the ADC and ePWM Modules .................... 15
12 C28x Pipeline Visualization ...................................................................... 16
13 Standard Operation for a C28x CPU Maskable Interrupt ............................. 17
14 CLB Integration in C2000 MCU Architecture .......................................... 18
15 CLB Configuration Tool in SysConfig ...................................................... 19
16 HRPWM Capability vs Traditional PWM Generation Methods .................. 20
17 Synchronous Boost Controller .................................................................. 22
18 Cycle by Cycle Trip Action of the COMP Module With Configurable Deadband....... 23
19 Industrial Servo Drive With T-Format Absolute Position Encoder Interface ....... 24
20 QepDiv Input and Output Diagram .......................................................... 25
21 Single-Ended Input Model ....................................................................... 26
22 ADC Post Processing Block on TMS320F2837xD ..................................... 31
23 CMPSS Visualization .............................................................................. 32
24 Filter and Demodulator Inside the C2000 SDFM ......................................... 33
25 SDFM Module With Both Primary and Secondary Filter Blocks on the TMS320F2837xD MCU ............................................................ 34
26 Full Duplex 3-Wire FSI Implementation .................................................... 35
27 FSI Skew Compensation .......................................................................... 36

List of Tables
1 TMU Supported Instructions Summary ....................................................... 6
2 Effective Flash Access Times With Prefetch Enable .................................... 11
3 Resolution for PWM vs HRPWM .............................................................. 20
4 Range of Acquisition Time Configuration (per Channel) .............................. 26
5 TMS320F28379D 16-Bit ADC Specifications ............................................. 29
6 Typical Resistor Tolerance Over Time and System Impact ......................... 30
7 Comparison of Fault Detection and Trip Methods ....................................... 32

Trademarks
Bosch is a trademark of Robert Bosch GmbH, Germany.
C2000, Code Composer Studio are trademarks of Texas Instruments.
1 C2000™ MCUs and Real-Time Control

A real-time control system is one that has time dependence to when it needs input to remain stable. This is counter to a batch processing type system in which data is manipulated by a processor and reported, but there is no critical system waiting for the data. Real-time control processors can perform batch type processing, but it is not typically their main use in a system. A simple visualization of a type of real-time control system is given in Figure 1. Thus far, only processing has been mentioned, but a fully realized controller in this case would be capable of measuring key parameters of the system (sensing), applying control algorithms to the incoming data (processing), and then affecting the system to achieve the desired change given by the control algorithm (actuation).

Figure 1. Simple Real-Time Control System
The amount of time that the system can be left alone (sometimes referred to as open loop) determines the speed at which the controller must perform all of the above functions. An important distinction here is that the raw speed of the processing block is not the only care about, but rather the MCU ecosystem as a whole contributes to the time it will take to react to different system events. Take a closer look at how the C2000 MCU is constructed to service these needs.

1.1 **Processing**

As seen in Figure 2, the C2000 MCU uses the C28x core as the main processing unit. This is a 32-bit floating point (single precision) core with dedicated instructions tailored to real-time control applications. Complementing the C28x core is the Control Law Accelerator (CLA), a 32-bit floating point co-processor capable of independent code execution increasing the system bandwidth versus a C28x core alone. There are both dual and single core implementations across the C2000 MCU family of devices.

1.2 **Actuation**

The actuation sub-system includes modules that will stimulate the system under control. Typically, this is done with a Pulse Width Modulation (PWM) output. This could also be the output of the on-chip Digital-to-Analog Converter (DAC) or just a General-Purpose Input/Output (GPIO) pin.

- **PWM** – Principle actuation module on the C2000 MCU. Responsible for driving the external Field Effect Transistors (FETs) that exist in most power electronics systems. Supports both standard and high resolution modes.
- **Buffered DAC** – 12-bit DAC capable of driving a defined external load. Typically used to create a bias voltage in the analog domain.
- **Configurable Logic Block** – Group of look up tables and state machine logic that operates on internal signal nodes in the hardware domain. Can be an endpoint or intermediary step to realize increased system performance.
1.3 Sensing

This sensing sub-system includes modules that translate the state of the external system under control (analog domain) into data usable by the MCU (digital domain). Often this is the work of the Analog-to-Digital Converters (ADCs) on the MCU, but could also be handled by comparators or demodulators for external ADCs. Other unit converters are included in this domain, such as quadrature encoders and time pulse measurement devices.

- **ADC** – Multiple 12 or 16-bit ADCs that are used primarily to convert the voltage or current (through a shunt) of the controlled system into the digital domain. Both an internal reference or external references are supported.
- **Comparator (COMP)** – Multiple on-chip comparators provide system protection as well as cycle by cycle PWM control by comparing a system voltage to an internal reference point (generated by the internal 12-bit DACs). Direct connection to the ePWM modules exist to change the output state as quickly as possible without need for CPU intervention.
- **Sigma Delta Demodulator** – On-chip logic used to decode the serial bit stream output from external sigma delta ADCs. Includes hardware threshold detection.
- **Quadrature Encoder Pulse Measurement (eQEP)** – Counts pulses from a variety of encoders to determine motor shaft position.
- **Time Capture (eCAP)** – Measures the time duration between external pulse events, useful for evaluating Hall Sensors. Both standard and high resolution modes are available.

1.4 Communications

An additional block that, while not essential in the real-time control of the system, is almost always needed from a system integration point of view. From serial data streams to multi-channel inputs, as well as industry standard options to proprietary formats, the communications sub-system supports a wide array of communications options.

- **Controller Area Network (CAN)** – The CAN module supports the Bosch™ CAN protocol standard.
- **External Memory InterFace (EMIF)** – Parallel data bus typically used to support connections to SDRAM as well as wide bus peripherals.
- **EtherCAT Slave Controller (EtherCAT)** – This module allows for the C2000 MCU to act as a slave node in an EtherCAT network.
- **Ethernet** – 10/100 Mbps Ethernet controller and physical interface for external communications across this bus.
- **Fast Serial Interface (FSI)** – 2 or 3 line simplex serial data transmit or receive. Designed to meet both the high speed (100Mbps) as well as the variable latency introduced when crossing an isolation boundary.
- **Inter-Integrated Circuit (I2C)** – Interface/controller for an I2C bus
- **Serial Peripheral Interface (SPI)** – Interface/controller for a standard Serial Peripheral Interface bus.
- **Universal Asynchronous Receiver/Transmitter (UART)** – Interface/controller for Universal Asynchronous Receiver Transmitter bus
- **Universal Serial Bus (USB)** – USB 2.0 MAC and PHY used to interface to standard USB network.

**NOTE:** Peripheral counts, as well as features, may vary from device to device. For a complete listing of the number of a peripherals on a device, see the data sheets referenced at the end of each section. For the feature sets supported on a given device, see the C2000 Real-Time Control Peripherals Reference Guide.
2 Processing Key Technologies

2.1 Accelerated Trigonometric Math Functions

2.1.1 Value Proposition

Trigonometric functions are used heavily in real-time control systems, both in power applications as well as motor control. Park Transforms (Figure 3), Space Vector Generation, and resolver angle are a few of these examples that rely on trigonometric math. The Trigonometric Math Unit (TMU) on C2000 MCUs enables an extended instruction set targeted at 32-bit floating-point trigonometry based calculations.

Figure 3. Park Transform

2.1.2 In Depth

Many common mathematical techniques in real-time control rely on the use of trigonometric functions: sine, cosine, and arc tangent are all examples. The TMU adds dedicated instructions to the C28x core for these functions as well as their inverse, that supersede the standard C library calls. As shown in Figure 4, an 85% cycle count improvement for the Park Transform is seen when using the TMU instructions vs the native C28x floating point compiled instruction set.

Figure 4. TMU improvement for Park Transform

There is also single instruction support for both square root as well as floating point division. These are often used in conjunction with the trigonometric functions previously listed. A full list of supported instructions and their cycle counts can be seen in Table 1. These instructions are inserted automatically by the C compiler on devices that have a TMU.

Table 1. TMU Supported Instructions Summary

<table>
<thead>
<tr>
<th>Operation</th>
<th>C Equivalent Operation</th>
<th>C28x Pipeline Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiply by 2*pi</td>
<td>a = b * 2pi</td>
<td>2 cycles + Sine/Cosine function</td>
</tr>
<tr>
<td>Divide by 2*pi</td>
<td>a = b / 2pi</td>
<td>2 cycles + Sine/Cosine function</td>
</tr>
<tr>
<td>Divide</td>
<td>a = b / c</td>
<td>5 cycles</td>
</tr>
<tr>
<td>Square Root</td>
<td>a = sqrt(b)</td>
<td>5 cycles</td>
</tr>
<tr>
<td>Sin Per Unit</td>
<td>a = sin(b*2pi)</td>
<td>4 cycles</td>
</tr>
<tr>
<td>Cos Per Unit</td>
<td>a = cos(b*2pi)</td>
<td>4 cycles</td>
</tr>
<tr>
<td>Arc Tangent Per Unit</td>
<td>a = atan(b)/2pi</td>
<td>4 cycles</td>
</tr>
<tr>
<td>Arc Tangent 2 and Quadrant Operation</td>
<td>Operation to assist in calculating ATANPU2</td>
<td>5 cycles</td>
</tr>
</tbody>
</table>
NOTE: While the C2000 MCUs listed below all have a TMU module, the C compiler used to generate the target code must have the correct options selected to utilize this HW. This is controlled in the Processor Options of the C2000 Compiler via the TMU support drop down, as well as using the "relaxed" setting for the floating point mode under C2000 Compiler → Optimizations. Specific TMU based functions can be called explicitly as inline functions in the C source if that is preferable to a global setting.

2.1.3 Device List

- TMS320F28378xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

2.1.4 Hardware Platforms and Software Examples

- TIDM-1007 Interleaved CCM Totem Pole Bridgeless Power Factor Correction (PFC) Reference Design
- TIDM-HV-1PH-DCAC Single-Phase Inverter Reference Design With Voltage Source and Grid Connected Modes
- TMDX1DDK379D C2000 DesignDRIVE Development Kit for Industrial Motor Control
- TMDSHVMTRINSPIN High Voltage Motor Control Kit with InstaSPIN-FOC and InstaSPIN-MOTION enabled Piccolo MCU for F280049C device lab7 and lab8

2.1.5 Documentation

- Enhancing the Performance Capabilities of the C2000™ MCU Family
2.2 Improving Control Performance With Nonlinear PID Control

2.2.1 Value Proposition

The nonlinear PID (NLPID) provides the ability to improve control loop performance beyond that normally available with linear controllers. The controller is available in the Digital Control Library (DCL), making it easy to integrate with existing C2000 code.

2.2.2 In Depth

Conventional linear controllers such as the Proportional, Integral, Derivative (PID) are widely used with digital power applications, including motor control and motion control. The nonlinear PID (DCL Training Video - Non-linear Control) provided with the DCL extends the performance of its linear counterpart by shaping the loop error using a nonlinear law. A nonlinear shaping block is introduced in series with each of the three controller paths as shown below.

![Figure 5. Non-Linear PID Block Diagram](image)

The shape and aggressiveness of nonlinear action are configurable via six additional controller parameters (two in each nonlinear block), which are typically tuned in an iterative fashion to optimize a transient response. Like other controllers in the DCL, the NLPID parameters can be updated safely using a shadow parameter set and an update function.
Figure 6 shows an example of the potential improvement in step response available from the use of nonlinear control action.

![Servo Response Plot](image)

**Figure 6. Comparison of Response Time Between Linear and Non-linear PID**

The NLPID executes with highest efficiency on devices equipped with the type 1 TMU (see Section 2.1), such as the F280025 device. These devices have CPU instructions which allow the nonlinear controller to be executed in 117 cycles compared with around 3,300 cycles without such instructions. This cycle efficiency allows the nonlinear PID to be used in high frequency applications such as switching power supplies and current control loops.

The DCL is packaged in C2000Ware, which is available for free download by C2000 users. The library includes a PID controller tuning guide to help users get the most from the NLPID controller.

### 2.2.3 Device List
- TMS320F28002x

### 2.2.4 Hardware Platforms and Software Examples
- Digital Control Library in C2000Ware
- F280025 controlCARD

### 2.2.5 Documentation
- DCL Training Video - Non-linear Control
2.3 Understanding Flash Memory Performance In Real-Time Control Applications

2.3.1 Value Proposition

In spite of the wait state penalty for flash memory accesses, the 128-bit-wide prefetch logic in association with the pipeline buffer makes sequential code execution performance equal to that of 0 wait state (WS) RAM. Allowing for common code discontinuities, most applications will run with an efficiency of approximately 80% relative to the code executing from RAM. In addition, there is a 128-bit cache on the data bus as well to improve data read performance. All of this performance enhancement comes with zero cycle-cost Error Code Correction (ECC) evaluation.

2.3.2 In Depth

Flash memory is a non-volatile memory which provides the advantage of retaining its content even after a power cycle. However, due to its physical construction flash memory is typically not as fast as volatile memories (SRAM, DRAM, and so forth). As a result, wait states are used in order to scale the MCU clock rate while accessing flash memory which can impact CPU performance. In order to greatly reduce this impact on performance, C2000 FMC’s (Flash Module Controller) read interface provides a prefetch mode (Figure 7). This mode significantly improves the performance of linear code, which typically makes up the majority of application code in real-time control systems.

Figure 7. C2000 Flash Prefetch Module

When enabled on C2000 MCUs, the module does a look-ahead prefetch (128-bit aligned) on linear address increments starting from the last instruction fetch address and stores it in a 128-bit wide by 2-level deep instruction prefetch buffer. This buffer can hold up to sixteen 16-bit instructions and will be continuously filled in the background by the prefetch mechanism as the CPU continues to use the already fetched instructions in the buffer. As a result, no wait states are incurred for each opcode fetch, which is a significant performance boost when compared to a one time wait-stated fetch. Wait states are incurred only when there is a program counter (PC) discontinuity such as a branch, function call, and so forth.
Table 2 provides some real-world examples of the code efficiency that can be expected on two different classes of C2000 flash devices.

### Table 2. Effective Flash Access Times With Prefetch Enable

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Device Properties</th>
<th>32-bit Float Math</th>
<th>16-bit if-then-else</th>
<th>Effective Performance 50/50 split</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMS320F2838xD/S</td>
<td>• 200 MHz CPU Clock • 50 MHz Flash Speed • 3 Wait States</td>
<td>• Flash Access Efficiency: 93% • Effective Performance: 188 MHz</td>
<td>• Flash Access Efficiency: 87% • Effective Performance: 174 MHz</td>
<td>181 MHz</td>
</tr>
<tr>
<td>TMS320F2837xD/S</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TMS320F28004x</td>
<td>• 100 MHz CPU Clock • 20 MHz Flash Speed • 4 Wait States</td>
<td>• Flash Access Efficiency: 84% • Effective Performance: 84 MHz</td>
<td>• Flash Access Efficiency: 84% • Effective Performance: 84 MHz</td>
<td>84 MHz</td>
</tr>
<tr>
<td>TMS320F28002x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** The number of flash wait states is always 1 less than the access time.
- $\text{FLASH\_wait\_states} = \text{FLASH\_access\_cycles} - 1$
- $\text{FLASH\_access\_cycles} = \text{round\_up} (\text{CPU MHz/FLASH MHz})$ For Example: $\text{round\_up} (165/50) = \text{round\_up} (3.3) = 4$

While the efficiency of program code executing from flash has been considered thus far, there also exists a 128-bit data cache. Users can enable this to increase the flash data read performance. When the CPU requests data from a flash address, the flash wrapper will store entire 128-bits of Flash data (aligned) in this cache instead of simply providing the requested address’s data to CPU. The CPU can access the remaining data in this cache without incurring any wait states. This data cache gets flushed and refilled when there is a cache miss.

Finally, there is also an Error Correction Code (ECC) value for each 64-bits of flash memory. The ECC is such that it provides for single bit error correction and dual bit error detection per 64-bits. The ECC is evaluated for correctness before the data is placed into the prefetch buffer with no impact to the access times/latencies mentioned previously. If an uncorrectable error is detected, a non-maskable interrupt is generated to halt normal code execution in parallel to the normal code execution of the CPU. You have the ability to set a threshold for correctable errors to trigger an interrupt to the C28x core as well.

#### 2.3.3 Supported Device List
- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

#### 2.3.4 Hardware Platforms and Software Examples
All code examples for the supported devices enable the prefetch buffer and ECC logic as part of their initialization routines inside the SysCtrl.c file.

#### 2.3.5 Documentation
- TMS320F2838xD/S Datasheet Flash Parameters
- TMS320F2837xD/S Datasheet Flash Parameters
- TMS320F28004x Datasheet Flash Parameters
- TMS320F28002x Microcontrollers Data Sheet
2.4 Flexible System Interconnect

2.4.1 Value Proposition

Three on-chip signal crossbars (X-bars): Input, Output, and ePWM provide the necessary mechanism in hardware to efficiently connect multiple subsystems across different control system implementations. Lower system latency, simpler PCB routing, and consistent timing are all key benefits.

2.4.2 In Depth

The inter-dependence of the Sensing and Actuation subsystems in a real-time control MCU is an obvious one. Whether it is incoming signals into the device, signals generated by on-chip logic (comparator, SDFM, ADC, and so forth), or outgoing signals; routing these signals in the system can be challenging at best. The on-chip X-bars provide a flexible mechanism to do that in hardware inside the MCU. There are three key benefits that these modules provide to the system:

- Simple routing of external signals into the chip: Any GPIO can go to multiple modules on-chip (Figure 8). For example, this allows the eCAP module to choose from any input pin as its source, or any input pin to go to the CPU/CLA as an external interrupt. This also gives flexibility in signal routing and layout of the Printed Circuit Board (PCB) since pins are not hard configured for a set group of functions.

Figure 8. Input X-Bar on the TMS320F2837xD MCU
• Lower system latency: For simple point-to-point transactions, there is no need for the main C28x CPU or the CLA to spend cycles routing signals from one domain to another when using the X-bars. Inside of each X-bar, there is also a simple logical OR (Figure 9) that allows combination of any of the inputs to the X-bar in hardware. This not only saves cycles for the processors, but allows flexibility of the system since the mux selections are all controlled in software and can be changed dynamically as needed.

Figure 9. Local Mux and Logical OR on the TMS320F2837xD MCU

• Consistent timing: Similar to the above, since there is no CPU involvement, the signal propagates at the system clock in real time as it happens and is not contingent on another block to allow the signal to pass (Figure 10). This results in more predictable and repeatable system behavior. Given that the nature of many of the signals is to pass into the actuation sub-system timing is especially critical.

Figure 10. X-Bar Sources and Destinations on the TMS320F2837xD MCU
2.4.3 Device List
- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

2.4.4 Hardware Platforms and Software Examples
- C2000 DesignDRIVE Development Kit for Industrial Motor Control

2.4.5 Documentation
- TMS320F2838x Microcontrollers Technical Reference Manual
- TMS320F2837xD Dual-Core Delfino Microcontrollers Technical Reference Manual

2.5 Increasing Control Loop Bandwidth With An Independent Processing Unit

2.5.1 Value Proposition
A primary concern in any control application is the time that elapses between sampling the system (sensing), applying the control function (processing), and applying the stimulus to the external system (actuation). The CLA was created specifically to address the need to minimize this time while increasing overall system throughput.

2.5.2 In Depth
The CLA found on the devices listed below is a fully parallel processor to the main C28x core. While the C28x core is a more traditional processor, executing instructions and servicing interrupts, the CLA is a task driven state machine. The CLA is a 32-bit floating point architecture.

Due to the nature of control systems, there are specific times when the sensing subsystem has new data to be processed. Advanced planning for these events with the other functions of an MCU can be difficult to time slice without introducing delay into both the system under control, but also any other functions the CPU has to perform.

As a task driven state machine, the CLA is constantly waiting in an idle state for an event, such as an ADC conversion, to process that data and actuate the system. Additionally, the CLA has full access to key control peripherals so it can fully realize the control system independent of the C28x CPU (Figure 11).
The above is beneficial for many reasons:

- There is little to no delay in processing the data, typically caused by the context switching of the main C28x core, and applying the new external stimulus to the system.
- There is no interruption or impact to the current C28x program execution.
- Potential to have parallel control systems running independently on the same MCU device.

The CLA has its own dedicated memory region for its code and shared memory for passing information between it and a C28x CPU in the system. The CLA is supported in Code Composer Studio™ by its own C compiler.

### 2.5.3 Device List

- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x

### 2.5.4 Hardware Platforms and Software Examples

- Valley switching boost power factor correction (PFC) reference design
- C2000 DesignDRIVE Development Kit for Industrial Motor Control

### 2.5.5 Documentation

- CLA Hands On Workshop
- CLA Usage in Valley Switching Boost Power Factor Correction (PFC) Reference Design
- CLA FAQ on E2E
- Enhancing the Performance Capabilities of the C2000™ MCU Family
2.6 Deterministic Program Execution

2.6.1 Value Proposition

One important aspect of any real-time control system is the consistency of program execution over time. Whether it is performance across the system update period or from multiple power ups over time, optimization of real-time systems relies on critical system events taking place at known points in time. The well defined 8 stage CPU pipeline of the C28x MCU, as well as the complementary behavior of its interrupt handling logic, provides this level of determinism.

2.6.2 In Depth

A processor that is not consistent in its behavior over time can cause perturbations to a real-time control system, either by not actuating the system in a timely manner or by sampling the state of the system at the incorrect point in time. Furthermore, real-time control systems are largely interrupt driven in their program flow. For this reason, a cache memory is undesirable since it would need to be discarded often whenever a program discontinuity occurs (in this case quite often).

Instead, large amounts of fast memory are desirable for program execution as well as a processor with a instruction pipeline that is deep enough to parallelize instructions, but also shallow enough to not incur large time penalties when discontinuities occur. The C28x CPU employs an 8 stage pipeline as shown in Figure 12. Once an instruction has entered the D2 phase of the pipeline, it cannot be stopped from full execution by an interrupt. Conversely, any instruction that is in a pre-D2 phase of the pipeline will be flushed when the incoming interrupt is received by the C28x core. Upon returning from the interrupt program, execution begins again with the F1 fetch stage. It is beyond the scope of this article to go deeper into the nuances of the pipeline, but those details are covered in the TMS320C28x DSP CPU and Instruction Set Reference Guide.

While you can see that the behavior of the program execution is repeatable over time, what about the behavior of the incoming interrupt? Figure 13 shows that this, too, is deterministic. Note that unless manually altered all other interrupt requests will pend until the current interrupt has been fully serviced. This is important in order to keep servicing of any interrupt consistent in the time domain once it begins. Both of these components of the C28x core help maintain deterministic code execution for the system.

Figure 12. C28x Pipeline Visualization
Interrupt request sent to CPU
Set corresponding IFR flag bit.

Interrupt enabled in IER?
Yes

Interrupt enabled by INTM bit?
Yes

Clear corresponding IFR bit.
Empty pipeline.
Increment and temporarily store PC.
Fetch interrupt vector.
Increment SP by 1.
Perform automatic context save.
Clear corresponding IER bit.
Set INTM and DBGM. Clear LOOP, EALLOW, and IDLESTAT.
Load PC with fetched vector.
Execute interrupt service routine.

Program continues

This sequence protected from interrupts

Figure 13. Standard Operation for a C28x CPU Maskable Interrupt

2.6.3 Device List
- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

2.6.4 Hardware Platforms and Software Examples
- F28388D controlCARD evaluation module
- C2000 MCU F28379D LaunchPad™ development kit
- C2000 MCU F280049C LaunchPad™ development kit
- F280025 controlCARD evaluation module

2.6.5 Documentation
- TMS320C28x CPU and Instruction Set Reference Guide
- C2000™ F2837xD Microcontroller Workshop
3 Actuation Key Technologies

3.1 On-Chip Hardware Customization

3.1.1 Value Proposition
The Configurable Logic Blocks (CLB) provide a mechanism for creating custom logic inside the chip. Enhancing existing subsystems, creating new subsystem and replacing off-chip logic circuits are all possible.

3.1.2 In Depth
The on-chip CLB provide a flexible mechanism to add personalized logic customization in hardware inside the MCU. Whether it is modifying an existing subsystem to satisfy an application specific need (Realizing Rotary Sensing Solutions Section 4.1), creating a completely new subsystem, or replacing external logic circuits, the CLB can accomplish all these tasks.

There are three key benefits that the CLB modules provide to the system:
1. Enhancing existing on-chip subsystems: The custom logic implemented inside the CLB modules can be inserted inside other on-chip subsystem, such as ePWM subsystem, to enhance the capability of the peripheral. Examples of this include creating a T-format absolute encoder ( ) interface or Pulse Train Output ( ).

2. Creating new subsystems: The CLB modules can be combined to form new subsystems, which could be a completely new peripheral that does not exist in C2000 MCUs, or replicate a C2000 MCU peripheral and create an extra subsystem, for example:
   a. Creating an auxiliary PWM module SW example
   b. Designing With the C2000 Configurable Logic Block as shown in CLB State Machine SW example

3. Replacing external logic: In some cases the CLB modules can be used to absorb external devices such as FPGA or CPLD which implement application specific customized logic. The CLB modules can also be used to replace external logic circuits. Examples of this are:
   a. Migrating from FPGA/CPLD to CLB
   b. Replacing external PWM protection circuit with custom logic in the CLB SW Example
While the physical implementation of the custom logic is controlled through uploaded memory values into the C2000 MCU, TI provides several GUI tools such as the CLB Configuration Tool (Figure 15) to both realize the logical implementation as well as verify the operation of the logic in simulation before it is used in the system.

![Figure 15. CLB Configuration Tool in SysConfig](image)

3.1.3 **Device List**
- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

3.1.4 **Hardware Platforms and Software Examples**
- LAUNCHXL-F28379D
- LAUNCHXL-F280049C
- Creating an auxiliary PWM module SW example
- CLB State Machine SW example
- Replacing external PWM protection circuit with custom logic in the CLB SW Example

3.1.5 **Documentation**
- CLB Programming Tool Training(Video)
- *Designing with the C2000™ Configurable Logic Block*
- *How to Migrate Custom Logic From an FPGA/CPLD to C2000™ Microcontrollers*
- *CLB Tool User's Guide*
3.2 Reducing Limit Cycling in Control Systems

3.2.1 Value Proposition

Limit cycling in a PWM controlled system refers to the in-ability for the PWM output to physically converge on the mathematical solution to the control law. This causes the PWM output to cycle about the true solution, resulting in instability in the control system. The High Resolution PWM (HRPWM) module on the C2000 MCU has the ability to modulate the PWM edge in 150 ps increments. This represents a 60 fold improvement over traditional PWM creation techniques based off the system clock rate (Figure 16) and can be used to realize a higher order of accuracy in PWM edge placement. A waveform’s period phase relationship to its complement, as well as deadband insertion time can all realize this high resolution benefit.

![Figure 16. HRPWM Capability vs Traditional PWM Generation Methods](image)

3.2.2 In Depth

All PWM-controlled power topologies are inherently bandwidth limited by the ability of the controller to place the PWM edge as close as possible to the mathematical solution of the control law. Whatever error exists from “rounding” the solution created in the form of an output PWM signal dictates the maximum efficiency that can be realized in the system.

In this sense, it may be helpful to think of the PWM as a type of DAC with a fixed resolution. Any error that results in the selection of the next available PWM edge placement would then be equivalent to quantization error term that is inherent to any DAC. Therefore, the minimum time step that can be achieved by a PWM module can be translated into “bits” of resolution of this equivalent DAC. As shown in Table 3, the increase of resolution of the C2000 MCU HRPWM vs a traditional PWM is very apparent, increasing the effective resolution by approximately 6 bits.

<table>
<thead>
<tr>
<th>PWM Freq (kHz)</th>
<th>Regular Resolution (PWM) 100 MHz EPWMCLK</th>
<th>High Resolution (HRPWM)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bits</td>
<td>% Error</td>
</tr>
<tr>
<td>20</td>
<td>12.3</td>
<td>0.02</td>
</tr>
<tr>
<td>50</td>
<td>11</td>
<td>0.05</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>0.1</td>
</tr>
<tr>
<td>150</td>
<td>9.4</td>
<td>0.15</td>
</tr>
<tr>
<td>200</td>
<td>9</td>
<td>0.2</td>
</tr>
<tr>
<td>250</td>
<td>8.6</td>
<td>0.25</td>
</tr>
<tr>
<td>500</td>
<td>7.6</td>
<td>0.5</td>
</tr>
<tr>
<td>1000</td>
<td>6.6</td>
<td>1</td>
</tr>
<tr>
<td>1500</td>
<td>6.1</td>
<td>1.5</td>
</tr>
<tr>
<td>2000</td>
<td>5.6</td>
<td>2</td>
</tr>
</tbody>
</table>
3.2.3 Device List

- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

3.2.4 Hardware Platforms and Software Examples

- TIDM-02002 Bidirectional CLLLC resonant dual active bridge for HEV/EV onboard charger
- TIDA-00961 Highly Efficient 1.6kW High Density GaN Based 1MHz CrM Totem-pole PFC Converter
- TIDA-010054 Bi-directional, dual active bridge reference design for level 3 electric vehicle charging stations
- C2000Ware HRPWM Example for TMS320F28388D
- C2000Ware HRPWM Example for TMS320F28379D
- C2000Ware HRPWM Example for TMS320F280049C

3.2.5 Documentation

- TMS320F2838xD Dual-Core Microcontrollers Technical Reference Manual
- TMS320F2837xD Dual-Core Microcontrollers Technical Reference Manual
- TMS320F28004x Dual-Core Microcontrollers Technical Reference Manual
- TMS320F28002x Dual-Core Delfino Microcontrollers Technical Reference Manual
- C2000 F2837xD Microcontroller 1-Day Workshop Section 1.6: Control Peripherals skip to 14:38 for HRPWM

3.3 Shoot Through Prevention for Current Control Topologies

3.3.1 Value Proposition

C2000 MCUs have the ability to implement current control techniques, such as Peak Current Mode Control (PCMC), in hardware using the on-chip comparators to control the PWM duty cycle. Variable deadband insertion has been added to the Type 4 PWM, enabling the ability to tune out potential shoot through, without the need for any CPU overhead.
3.3.2 In Depth

For better power efficiency, many DC-DC systems implement a synchronous boost controller, where the secondary switch replaces the feed forward diode that exists in a regular boost controller (Figure 17). Peak Current Mode Control is one of the more common methods used to control this topology and the C2000 MCU has some unique features that allow it to implement this type of control very efficiently.

![Figure 17. Synchronous Boost Controller](image)

The addition of the second FET demands precise control of the ON/OFF time of the primary and secondary switch relative to one another. If the switches are in the “ON” state at the same time, there is a direct path to ground for the active current to flow, which is not only inefficient but also potentially harmful to the lifetime of the FET switches.

An accurate way to implement this type of system is to have the comparator monitor the inductor current and actuate the FETs when the current exceeds a predefined threshold. Ideally, when one FET is switched ON, the other FET can be switched OFF at the same time. However, due to switch mis-match and board propagation delays, simultaneous switching from the controller likely will not result in simultaneous switching at the FETs, which creates the shoot through mentioned earlier. While techniques in software can help hold off the switching of the secondary switch to avoid this condition, these can be challenging to implement with the various hardware interdependencies coupled with the time constraints of the control loop.
The C2000 MCU has implemented programmable deadband control, derived from the comparator output itself, to prevent this condition while keeping the C28x CPU unloaded (Figure 18). This allows a complete PCMC solution to be realized outside of the CPU domain once initialized. This logic exists on all PWM modules on a given device, allowing multiple stages to have different deadbands, such as a Phase Shifted Full Bridge, where there are multiple switching pairs.

![Figure 18. Cycle by Cycle Trip Action of the COMP Module With Configurable Deadband](image)

### 3.3.3 Device List
- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

### 3.3.4 Documentation
- [TMS320F2838xD Dual-Core Microcontrollers Technical Reference Manual](#) (for more information, see the Action-Qualifier (AQ) Submodule and Dead-Band Generator (DB) Submodule sections of the Enhanced Pulse Width Modulator (ePWM) chapter)
- [TMS320F2837xD Dual-Core Microcontrollers Technical Reference Manual](#) (for more information, see the Action-Qualifier (AQ) Submodule and Dead-Band Generator (DB) Submodule sections of the Enhanced Pulse Width Modulator (ePWM) chapter)
- [TMS320F28004x Microcontrollers Technical Reference Manual](#) (for more information, see the Action-Qualifier (AQ) Submodule and Dead-Band Generator (DB) Submodule sections of the Enhanced Pulse Width Modulator (ePWM) chapter)
- [TMS320F28002x Piccolo Microcontrollers Technical Reference Manual (SPRuin7)](#) (for more information, see the Action-Qualifier (AQ) Submodule and Dead-Band Generator (DB) Submodule sections of the Enhanced Pulse Width Modulator (ePWM) chapter)
- F2837xD MCU 1 day Workshop Control Peripherals
4 Sensing Key Technologies

4.1 Realizing Rotary Sensing Solutions

4.1.1 Value Proposition

As mentioned in the previous article: On-Chip Hardware Customization (Section 3.1), the C2000 Configurable Logic Block (CLB) technology enables systems designers to implement new logic within the C2000 device and eliminate external circuitry. A specific example for a real time control system is an integrated interface to digital rotary encoders; using the CLB, developers have the freedom to integrate an industry-standard encoder communications protocol or an application-specific customized protocol inside the C2000 MCU without additional external circuitry.

4.1.2 In Depth

A digital rotary encoder is a device that converts the position of a shaft to a digital signal. There are two main types of encoders:

1. **Absolute Encoders** (Figure 19): The output of an absolute encoder indicates the current angular position in a message sent back to the master as defined by a particular protocol.

2. **Incremental Encoders**: The output of an incremental encoder provides information in a train of modulated pulses which is typically further processed by the system master into information such as speed, distance and position.

C2000 CLB technology enables an integrated solution to interface to the most popular digital rotary position encoders, eliminating the necessity for external field-programmable gate arrays (FPGAs) or application-specific integrated circuits (ASICs). The PositionManager BoosterPack plug-in module is a flexible low voltage platform intended for evaluating various encoder interfaces and designed to work with multiple C2000 LaunchPad development kits such as the LAUNCHXL-F28379D or LAUNCHXL-F280049C.

Below are examples of encoders that are available for evaluation today as part of the Motor Control SDK software package. Future updates are planned to add examples for both the BiSS-C and EnDAT22 protocols.

- **T-Format Absolute Encoder Interface**:
  
  The Tamagawa T-Format protocol is a popular digital, bidirectional interface for absolute encoders. The easy-to-use library and example software delivered with TIDM-1011 demonstrates Tamagawa's T-Format standard. In this example, the T-Format absolute encoder interface is integrated into the C2000 using on-chip resources such as the CLB, SPI, and GPIO as shown in Figure 19. This TI Design includes the following features:
  
  - A T-Format encoder interface library and CRC library which implements protocol commands
  - A demonstration project (with full source code access) to exercise the T-Format commands
  - Fully supported by the C2000 CLB Tool integrated inside Code Composer Studio

![Figure 19. Industrial Servo Drive With T-Format Absolute Position Encoder Interface](image-url)
• T-Format Evaluation in High-Bandwidth Current Loop Applications:

With the recent C2000 MCUs such as TMS320F2837x and TMS320F28004x, it is possible to implement Fast Current Loop (FCL) algorithms that provide a high current loop bandwidth with the same external hardware as used in classical Field Oriented Control (FOC) methods. TI has developed the FCL algorithm on these MCUs and implemented it on the DesignDRIVE IDDK platform. The T-Format encoder interface has been integrated into an evaluation implementation of FCL algorithms on C2000 devices.

Quick Response Control of PMSM Using Fast Current Loop studies the frequency response analysis of current loops in real time and also verifies the interface logic for the T-format encoder interface implemented in TIDM-1011. The position loop in this example can be closed using a QEP encoder or a T-format encoder and FCL can be implemented in both cases.

• Pulse Train Output (PTO) QepDiv and PulseGen:

Incremental rotary encoders output a pulse train to indicate that the shaft being monitored has moved. The system master typically processes this pulse train to determine information such as speed, distance and position. In the QepDiv implementation, position information is sent from the encoder to the Enhanced Quadrature Encoder Pulse (eQEP) module on a C2000 MCU. In the PulseGen case a custom pulse stream is generated to meet the system needs. The PTO-QepDiv example demonstrates how the CLB can be used to generate a divided pulse stream from these eQEP inputs as shown in Figure 20. The divided pulse stream can then be sent to another device in the system. Both the QepDiv and PulseGen examples are documented in the C2000 Position Manager PTO API Reference Guide.

Figure 20. QepDiv Input and Output Diagram

4.1.3 Device List

• TMS320F2838xD/S
• TMS320F2837xD/S
• TMS320F2807x
• TMS320F28004x
• TMS320F28002x

4.1.4 Hardware Platforms and Software Examples

• Tamagawa T-Format Absolute-Encoder Master Interface Reference Design for C2000™ MCUs
• C2000 Position Manager PTO API Reference Guide
• MotorControl software development kit (SDK) for C2000 MCUs
• Position Manager BoosterPack (BOOSTXL-POSMGR)
• C2000™ DesignDRIVE Development Kit for Industrial Motor Control (TMDXIDDK379D)

4.1.5 Documentation

• Training: How the C2000 Configurable Logic Block (CLB) tool integrates custom logic in my design
• CLB Tool User’s Guide
• Designing With the C2000 Configurable Logic Block (CLB)
• How to Migrate Custom Logic From an FPGA/CPLD to C2000 Microcontrollers
• Fast Current Loop Library
4.2 Optimizing Acquisition Time vs Circuit Complexity for Analog Inputs

4.2.1 Value Proposition

Control systems have the need to interface with a variety of feedback and monitoring sources. Signal sources in these systems differ in their ability to drive a capacitive input circuit like those typically found in the sample-and-hold (S+H) input circuit of an analog-to-digital converter (ADC). The ADCs on C2000 devices allow the acquisition time of the S+H to be individually configured for each input channel over a wide range. This allows the system to simultaneously interface with a mix of high-performance and low-cost signal sources.

4.2.2 In Depth

The inputs of an ADC are typically modeled as a switched capacitor circuit where the hold capacitor inside the ADC, $C_h$, needs to be charged from an unknown voltage to a value close to the input voltage during the acquisition time. An example, taken from the TMS320F2837xD device, is shown in Figure 21.

The required acquisition time for charging $C_h$ is determined by the external impedance of passive components, bandwidth of any buffers or sensors, the internal ADC input parasitics, and the resolution of the ADC.

The system designer can make a variety of trade-offs with respect to external circuit cost and complexity vs settling speed, for example:

- **Adding/upgrading the op-amp buffer driving the ADC inputs**: Lowering acquisition time through better charge transfer to the sample and hold capacitor inside the ADC
- **Increasing the amount of resistance and/or capacitance seen by the ADC input**: Helps reduce noise by adding additional low-pass filtering at the expense of a longer acquisition time
- **Tolerating less accuracy**: Alternatively, using a smaller acquisition window to decrease the sampling time, at the expense of accuracy/resolution.

With all the above possible trade-offs, it is difficult to select a single acquisition time that is appropriate for all analog inputs in the system. C2000 ADCs allow a separate acquisition window to be selected for each channel, giving the system designer a great deal of flexibility to make whatever speed vs signal conditioning circuit cost vs accuracy trade-offs they would like. The acquisition window (controlled by the ACQPS field of the ADC SOC configuration register) can also be configured over a wide range of values and with a small step size as shown in Table 4.

**Table 4. Range of Acquisition Time Configuration (per Channel)**

<table>
<thead>
<tr>
<th>C2000 MCU</th>
<th>Device SYCLK</th>
<th>Minimum S+H Time</th>
<th>Maximum S+H Time</th>
<th>S+H Time Configuration Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMS320F28004x and TMS320F28002x</td>
<td>100 MHz</td>
<td>80 ns</td>
<td>5.1 µs</td>
<td>10.00 ns</td>
</tr>
<tr>
<td>TMS320F2807x</td>
<td>120 MHz</td>
<td>75 ns</td>
<td>4.3 µs</td>
<td>8.33 ns</td>
</tr>
<tr>
<td>TMS320F2837xD and TMS320F2837xS</td>
<td>200 MHz</td>
<td>75 ns</td>
<td>2.6 µs</td>
<td>5.00 ns</td>
</tr>
</tbody>
</table>

There are a variety of ways to model the ADC input, Texas Instruments offers free tools to help design the ADC input driver circuit as well as instructional videos on proper front end component selection.
4.2.3 Supported Devices
- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

4.2.4 Hardware Platforms and Software Examples
- F28388D controlCARD
- F28379D controlCARD
- F280049C controlCARD
- F280025 controlCARD
- F2838xD ADC SW Example
- F2837xD ADC SW Example
- F28004x ADC SW Example

4.2.5 Documentation
- TMS320F2838x Microcontrollers Technical Reference Manual (for more information, see Choosing an Acquisition Window Duration section)
- TMS320F2837xD Dual-Core Microcontrollers Technical Reference Manual (for more information, see Choosing an Acquisition Window Duration section)
- TMS320F28004x Microcontrollers Technical Reference Manual (for more information, see Choosing an Acquisition Window Duration section)
- TMS320F28002x Microcontrollers Technical Reference Manual (for more information, see Choosing an Acquisition Window Duration section)
- SAR ADC Input Driver Design
- TI Precision Labs - ADCs: Introduction to SAR ADC Front-End Component Selection
4.3 Accurate Digital Domain Representation of Analog Signals

4.3.1 Value Proposition

Many MCUs have integrated ADCs as part of their sensing subsystem. The ability of the ADC to accurately convert the analog domain to the digital space is one of the most crucial aspects of the MCU in order to realize a proper control system. The data sheet specifications for a C2000 MCU ADC are such that performance in the system can be properly evaluated prior to system implementation.

4.3.2 In Depth

The first step when selecting an MCU for a real-time control system is a relatively straightforward process; comparing the components of the MCU to the system needs. There are questions of memory size, CPU speed, communications standards used, analog content, number of I/Os, and so forth. When looking at the fit for an analog module like the ADC, it can appear straightforward to base the decision on sampling rate, number of inputs, and bit level. In practice, however, there is much more to this decision.

Too often ADC selection is based solely on the top level specifications, only to realize during development there are limitations to the system performance due to the ADC itself:

- Will the system be using the analog inputs for frequency analysis? Then, AC specifications like SNR and THD become important to consider when picking an MCU with an on-chip ADC.
- Is overall accuracy a key care about? Looking at the DC specifications like INL, Gain, and Offset are key parameters to consider.

A quick summary of ADC specifications and their relevance to the system:

- **AC Specifications**: Parameters related to how accurately the converter can resolve the fundamental frequency tone of a signal from other noise sources. Includes SNR, SINAD, THD, and SFDR all expressed in dB. Also includes ENOB, which is the SINAD translated into number of bits. Typically SINAD and ENOB based on SINAD are considered when choosing an ADC, the importance will vary depending on the end application.

- **DC Specifications**: Parameters related to the accuracy of the converter as it applies to representing an analog input in the digital domain. Includes Gain, Offset, DNL, and INL. The weighted summation of the Gain, Offset, and INL are often referred to as "Total Unadjusted Error" (Equation 1). This equation is typically used to determine the real-world impact of these parameters on the accuracy of a conversion.

\[
\sum \sqrt{(Err_{\text{gain}})^2 + (Err_{\text{offset}})^2 + (Err_{\text{INL}})^2}
\]

where
- \(Err_{\text{gain}}\) is the maximum gain error of the ADC in LSBs
- \(Err_{\text{offset}}\) is the maximum offset error ADC in LSBs
- \(Err_{\text{INL}}\) is the maximum INL error of the ADC in LSBs

An example of how the C2000 ADC is specified and the parameters can be seen in Table 5, a dynamic link to this same table in the data sheet is located here.

One final aspect of all the parameters that C2000 devices list in the data sheet is what is implied by the inclusion of the parameter itself. For parameters that have a MIN/MAX, these are assured specs over the full operational range and lifetime of the device. The typical (TYP) column is also significant for all parameters, as it represents the mean performance of a parameter across its operational range.
### Table 5. TMS320F28379D 16-Bit ADC Specifications

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Test Conditions</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC conversion cycles</td>
<td></td>
<td>29.6</td>
<td>31</td>
<td></td>
<td>ADCCLKs</td>
</tr>
<tr>
<td>Power-up time (after setting ADCPWNZ to first conversion)</td>
<td></td>
<td></td>
<td>500</td>
<td>µs</td>
<td></td>
</tr>
<tr>
<td>Gain error</td>
<td></td>
<td>–64</td>
<td>±9</td>
<td>64</td>
<td>LSBs</td>
</tr>
<tr>
<td>Offset error</td>
<td></td>
<td>–16</td>
<td>±9</td>
<td>16</td>
<td>LSBs</td>
</tr>
<tr>
<td>Channel-to-channel gain error</td>
<td></td>
<td>±3</td>
<td></td>
<td></td>
<td>LSBs</td>
</tr>
<tr>
<td>Channel-to-channel offset error</td>
<td></td>
<td>±3</td>
<td></td>
<td></td>
<td>LSBs</td>
</tr>
<tr>
<td>ADC-to-ADC gain error</td>
<td>Identical VREFHI and VREFLO for all ADCs</td>
<td>±6</td>
<td></td>
<td></td>
<td>LSBs</td>
</tr>
<tr>
<td>ADC-to-ADC offset error</td>
<td>Identical VREFHI and VREFLO for all ADCs</td>
<td>±3</td>
<td></td>
<td></td>
<td>LSBs</td>
</tr>
<tr>
<td>DNL</td>
<td></td>
<td>&gt; –1</td>
<td>±0.5</td>
<td>1</td>
<td>LSBs</td>
</tr>
<tr>
<td>INL</td>
<td></td>
<td>–3</td>
<td>±1.5</td>
<td>3</td>
<td>LSBs</td>
</tr>
<tr>
<td>SNR</td>
<td>VREFHI = 2.5 V, fIN = 10 kHz</td>
<td>87.6</td>
<td></td>
<td></td>
<td>dB</td>
</tr>
<tr>
<td>THD</td>
<td>VREFHI = 2.5 V, fIN = 10 kHz</td>
<td>–93.5</td>
<td></td>
<td></td>
<td>dB</td>
</tr>
<tr>
<td>SFDR</td>
<td>VREFHI = 2.5 V, fIN = 10 kHz</td>
<td>95.4</td>
<td></td>
<td></td>
<td>dB</td>
</tr>
<tr>
<td>SINAD</td>
<td>VREFHI = 2.5 V, fIN = 10 kHz</td>
<td>86.6</td>
<td></td>
<td></td>
<td>dB</td>
</tr>
<tr>
<td>ENOB</td>
<td></td>
<td>14.1</td>
<td></td>
<td></td>
<td>bits</td>
</tr>
<tr>
<td></td>
<td>VREFHI = 2.5 V, fIN = 10 kHz, single ADC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>VREFHI = 2.5 V, fIN = 10 kHz, synchronous ADCs</td>
<td>14.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>VREFHI = 2.5 V, fIN = 10 kHz, asynchronous ADCs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSRR</td>
<td>VDDA = 3.3-V DC + 200 mV DC up to Sine at 1 kHz</td>
<td>77</td>
<td></td>
<td></td>
<td>dB</td>
</tr>
<tr>
<td>PSRR</td>
<td>VDDA = 3.3-V DC + 200 mV Sine at 800 kHz</td>
<td>74</td>
<td></td>
<td></td>
<td>dB</td>
</tr>
<tr>
<td>CMRR</td>
<td>DC to 1 MHz</td>
<td>60</td>
<td></td>
<td></td>
<td>dB</td>
</tr>
<tr>
<td>VREFHI input current</td>
<td></td>
<td>190</td>
<td></td>
<td>µA</td>
<td></td>
</tr>
<tr>
<td>ADC-to-ADC isolation</td>
<td>VREFHI = 2.5 V, synchronous ADCs</td>
<td>–2</td>
<td></td>
<td>2</td>
<td>LSBs</td>
</tr>
<tr>
<td></td>
<td>VREFHI = 2.5 V, asynchronous ADCs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 4.3.3 Device List
- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

### 4.3.4 Hardware Platforms and Software Examples
All controlCARDs for their specific C2000 MCU have been verified to reproduce the DS specifications for the on-chip ADC
- TMDSNCND28388D
- TMDSNCND28379D
- TMDSNCND280049C
- TMDSNCND280025
4.4 Resolving Tolerance and Aging Effects During ADC Sampling

4.4.1 Value Proposition

The ADC result is often manipulated mathematically before its use in the control law of a given system. This is typically done with some additional operations by the CPU, adding increased latency to the system as well as loading the CPU for such operations. The C2000 MCU has the ability to correct for this in hardware with no CPU overhead and no impact to ADC sample rate.

4.4.2 In Depth

Prior to using the ADC result in control calculations, it is often necessary to remove any known offset introduced by external factors such as component tolerances or layout differences (Table 6). While aspects of the above issues can be partially addressed through PCB layout or choosing higher tolerance/stable resistors there are always deviations from the ideal.

Table 6. Typical Resistor Tolerance Over Time and System Impact

<table>
<thead>
<tr>
<th>Life Cycle Stage</th>
<th>Total Tolerance</th>
<th>Associated 12-Bit Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purchase</td>
<td>±0.05%</td>
<td>±2 LSBs</td>
</tr>
<tr>
<td>Post Assembly</td>
<td>±0.5%</td>
<td>±20 LSBs</td>
</tr>
<tr>
<td>Post Storage/Moisture</td>
<td>±0.75%</td>
<td>±30 LSBs</td>
</tr>
<tr>
<td>Temp Coeff and EOL</td>
<td>±1.00%</td>
<td>±40 LSBs</td>
</tr>
</tbody>
</table>

C2000 MCUs implement an integrated hardware block to correct up to a 10-bit signed value co-incident to the ADC conversion process, saving valuable cycles in the system. The cycle value to the system is effectively doubled, as the ADC sample rate is maintained and no CPU cycles are used to perform the correction. Saturation is built in as well.
For the implementation of the offset correction in addition to the other modules included in the ADC Post Processing Block, see Figure 22.

**Figure 22. ADC Post Processing Block on TMS320F2837xD**

4.4.3 Device List

- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

4.4.4 Hardware Platforms and Software Examples

- TMDSIDDK379D

4.4.5 Documentation

- TMS320F2838xD Dual-Core Microcontrollers Technical Reference Manual - see the Post-Processing Blocks section in the Analog-to-Digital Converter (ADC) chapter
- TMS320F2837xD Dual-Core Microcontrollers Technical Reference Manual - see the Post-Processing Blocks section in the Analog-to-Digital Converter (ADC) chapter
- TMS320F28004x Microcontrollers Technical Reference Manual - see the Post-Processing Blocks section in the Analog-to-Digital Converter (ADC) chapter
- TMS320F28002x Microcontrollers Technical Reference Manual - see the Post-Processing Blocks section in the Analog-to-Digital Converter (ADC) chapter
4.5 Fast Detection of Over and Under Currents and Voltages

4.5.1 Value Proposition

Every control system can experience random events that can cause damage to the system. Fast detection and reaction to these events is critical to keeping the system safe and in good working condition. The on-chip comparators can detect and react to these events in a fraction of the time that it would take for an ADC and processor.

4.5.2 In Depth

Fault detection and reaction is important in most systems, not only for avoiding an undefined output, but for preventing damage to components both on and off the main Printed Circuit Board (PCB). The speed at which the fault detection takes place, as well as the final FET output state change, is critical to the system. A dedicated subsystem that ties together the analog and digital domains has been implemented on the C2000 MCU for handling this requirement: the Comparator Subsystem or CMPSS (Figure 23).

There are up to eight CMPSS modules on each C2000 MCU, with internal DACs that give the inverting/comparison detection level for the line that is being monitored. As shown in Figure 23, each CMPSS module has two comparators for simultaneous high and low detection. Using the CMPSS has several advantages over using the ADC for fault detection:

- **System Overhead**: Using the CMPSS to monitor a pin is an essentially zero overhead operation after the initial setup. The pin is always monitored against the comparison value until disabled. Other techniques would require periodic ADC conversions and threshold checking.

- **Latency**: While the ADC sampling rate can be simply factored into the period of the control loop, there is not a deterministic constant to a fault condition. As such there will be an inherent delay to detect the fail, both from a point of sample to the conversion time of the ADC itself. The comparator has no such trigger requirement or sample time, it is continuously monitoring the analog signal.

- **Dedicated PWM Trip Zone input**: The output of every CMPSS module can be tied directly into the Trip Zone of any PWM, and the action when the signal is received is configurable in software. This means there is no software overhead as there would be in processing an ADC ISR to then create the action to the PWM in software.

- **No clock dependence**: Since by definition a comparator is a purely analog domain circuit there are not clock dependencies to the changing state of its output based on the input. The C2000 MCU has carried this forward to give a asynchronous path from the comparator to the PWM. This allows for the fastest possible time from fault detection to pin state change, in addition to removing any clock dependence (Table 7).

### Table 7. Comparison of Fault Detection and Trip Methods

<table>
<thead>
<tr>
<th>Sampling Method</th>
<th>Sample Time (min)</th>
<th>Result Ready (min)</th>
<th>Latch and Change PWM Pin (@200 MHz SysClk)</th>
<th>Total Time From Fault To Trip</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-bit ADC</td>
<td>75 ns</td>
<td>260 ns</td>
<td>approximately 100 ns (inc ISR)</td>
<td>435 ns</td>
</tr>
<tr>
<td>12-bit ADC w/PPB</td>
<td>75 ns</td>
<td>260 ns</td>
<td>10 ns</td>
<td>355 ns</td>
</tr>
<tr>
<td>CMPSS</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>60 ns</td>
</tr>
</tbody>
</table>
• Simultaneous high and low detection: Each input to a CMPSS module routes the signal to two physically independent comparators that give the ability to detect both overshoot and undershoot at the same time.

4.5.3 Device List

- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

4.5.4 Hardware Platforms and Software Examples

The following kits implement the CMPSS to detect out of range current/voltage events

- TIDM-02002 Bi-Directional CLLLC Resonant Dual Active Bridge Reference Design for HEV/EV Onboard Charger
- TIDM-1022 Valley Switching Boost Power Factor Corrector
- TMDXIDDK379D C2000 Design DRIVE Development Kit for Industrial Motor Control

4.5.5 Documentation

- Comparator Specifications in the TMS320F28379D Data Sheet
- TMS320F2837xD Dual-Core Microcontrollers Technical Reference Manual (for more information, see the Comparator Subsystem (CMPSS) chapter)

4.6 Smart Sensing Across An Isolation Boundary

4.6.1 Value Proposition

High voltage control systems often make use of a sigma-delta ADC in order to easily pass the analog information of the high voltage domain to the lower voltage domain where the MCU exists. Before the data stream can be used, it must be processed by a filter and demodulator. This logic exists on the C2000 MCU and is called the Sigma Delta Filter Module (SDFM) module. The high and low comparators inside the SDFM on a C2000 MCU can actuate the PWMs without CPU intervention saving valuable time to better control the system.

4.6.2 In Depth

A sigma-delta type ADC is by design an over-sampling architecture. The ADC itself is a single bit design that over-samples the signal of interest to produce a higher order output. The modulation of this type of converter results in a binary output and, hence, a serial data stream. This serial data stream is then sent to the filter/demodulator on the C2000 MCU for re-construction into a higher bit order digital representation of the sampled signal (Figure 24).

At this point an interrupt is generated to the MCU, informing the other domains there is new data to process and act upon.
The SDFM module on the C2000 MCU is unique in that not only can the converted data be read post filtering, but the PWMs can be switched based on this data automatically. Each SDFM module contains four channels. Inside each channel exists two filters: a primary filter that produces the SDFM data and a secondary filter containing both high and low limit comparators (Figure 25). This allows the system to control the PWM signals without waiting for CPU intervention, resulting in both lower latency as well as lower overall CPU utilization.

![Figure 25. SDFM Module With Both Primary and Secondary Filter Blocks on the TMS320F2837xD MCU](image)

### 4.6.3 Device List
- TMS320F2838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x

### 4.6.4 Hardware Platforms and Software Examples
- TMDSCNCD28388D controlCARD evaluation module
- LAUNCHXL-F28379D LaunchPad
- SDFM_filter_sync_cpuread SW example for F2838xD/S
- SDFM_pwm_sync_cpuread SW example for the F2838xD/S

### 4.6.5 Documentation
- *TMS320F2838x Microcontrollers Technical Reference Manual* (for more information, see the *SDFM* chapter)
- Real-time controllers get new connectivity capabilities
5 Communications

5.1 Distributed Real-Time Control Across an Isolation Boundary

5.1.1 Value Proposition

Many real-time control systems rely on external communications for vital system information either as inputs into the control loop or as monitors to the system. Due to the time critical nature of any operation in these systems both the absolute speed as well as the integrity of the data is important. C2000 MCU’s Fast Serial Interface (FSI) fulfills this need offering higher throughput compared to other serial communication peripherals, up to 200 Mbps, along with additional features in HW that add data integrity using only a few wires.

5.1.2 In Depth

There are a number of serial communication peripherals to choose from when designing a multi-device real-time control system. With processors needing to pass critical data between each other within very short periods of time, latency is a primary concern for the system designer.

The FSI physical interface consists of three wires, a clock and two data signals, where one of the data signals is optional (see Figure 26). Data is transferred on both the rising and falling edge which permits a 50 MHz maximum FSI clock frequency to transfer data at 100 Mbps with two wires (CLK and D0) and 200 Mbps with three wires (CLK, D0, and D1). The high through-put along with defined data packets (frames) that contain limited header and footer allows data to be transferred between devices with very little latency. The FSI module consists of independent transmitter and receiver cores which allow for simultaneous full speed communications in both directions with no concept of a master or slave. A real-time system using FSI for distributed control is showcased in Distributed Multi-axis Servo Drive Over Fast Serial Interface (FSI) Reference Design.
Features that FSI offers over other commonly used communications peripherals include:

- Hardware implemented CRC at both the transmitter and receiver side eliminating the CPU overhead of a SW implementation
- Delay line control at the receiver module to compensate for channel-to-channel skew
- Line break detection using ping and data frame watchdogs
- FSI protocol has no concept of masters and slaves enabling devices to send feedback at any time without the master device having to make a request
- High bit rate with low signal count reduces the amount of isolators needed in the system

There are a number of system topologies which have components operating on both the “hot” (high voltage) and “cold” (low voltage) sides of the system that must communicate with each other. In this case digital isolators are used to bring data across an isolation barrier and the potential skew between signals that cross the isolation boundary can prove difficult to predict across many units of production. Even in systems without isolation, skew could be introduced by unequal signal trace lengths.

The delay line control feature at the receiver makes FSI well suited for this application as it can actively compensate for this skew (Figure 27) by adding delay to the individual FSI signals. See the Fast Serial Interface (FSI) Skew Compensation for more information on this differentiated feature. Also see the TMDSFSIADAPEVM for hardware evaluation of FSI with digital isolators.

![Figure 27. FSI Skew Compensation](image)

**NOTE:** Data is always transmitted and received on both the rising and falling edges of the FSI clock. While only one data line is show in Figure 27 there is option of second data line in all FSI implementations

### 5.1.3 Device List

- TMS320F2838xD/S
- TMS320F28004x
- TMS320F28002x

### 5.1.4 Hardware Platforms and Software Examples

- TMDSFSIADAPEVM
- LAUNCHXL-F280049C
- TIDM-02006

### 5.1.5 Documentation

- *Fast Serial Interface (FSI) Skew Compensation*
- *Using the Fast Serial Interface (FSI) With Multiple Devices in an Application*
6 References

6.1 Device List

- TMS320F3838xD/S
- TMS320F2837xD/S
- TMS320F2807x
- TMS320F28004x
- TMS320F28002x

6.2 Hardware/Software Resources

- TIDM-1007 Interleaved CCM Totem Pole Bridgeless Power Factor Correction (PFC) Reference Design
- TIDM-HV-1PH-DCAC Single-Phase Inverter Reference Design With Voltage Source and Grid Connected Modes
- TMDXI/DDK379D C2000 DesignDRIVE Development Kit for Industrial Motor Control
- TMDSSVMTRINSPIN High Voltage Motor Control Kit with InstaSPIN-FOC and InstaSPIN-MOTION enabled Piccolo MCU for F280049C device lab7 and lab8
- Vienna Rectifier-Based Three Phase Power Factor Correction Reference Design Using C2000 MCU
- TIDM-02002 Bidirectional CLLLC resonant dual active bridge for HEV/EV onboard charger
- TIDA-00961 Highly Efficient 1.6kW High Density GaN Based 1MHz CrM Totem-pole PFC Converter
- F28388D controlCARD evaluation module
- C2000 MCU F28379D LaunchPad™ development kit
- C2000 MCU F280049C LaunchPad™ development kit
- C2000 DesignDRIVE Development Kit for Industrial Motor Control
- Valley switching boost power factor correction (PFC) reference design
- TMDSIDDK379D
- TMDSCNCD28379D
- TMDSCNCD280049C
- TIDM-1022 Valley Switching Boost Power Factor Corrector

6.3 Documentation

- Texas Instruments: Enhancing the Performance Capabilities of the C2000™ MCU Family
- Texas Instruments: TMS320C28x CPU and Instruction Set Reference Guide
- Texas Instruments: TMS320F28002x Microcontrollers Data Sheet
- Texas Instruments: Designing with the C2000™ Configurable Logic Block
- Texas Instruments: How to Migrate Custom Logic From an FPGA/CPLD to C2000™ Microcontrollers
- Texas Instruments: CLB Tool User's Guide
- Texas Instruments: Quick Response Control of PMSM Using Fast Current Loop
- Texas Instruments: Fast Current Loop Library
- Texas Instruments: C2000 Position Manager PTO API Reference Guide
References

- Texas Instruments: Distributed Multi-axis Servo Drive Over Fast Serial Interface (FSI) Reference Design
- Texas Instruments: Fast Serial Interface (FSI) Skew Compensation
- Texas Instruments: Using the Fast Serial Interface (FSI) With Multiple Devices in an Application
- Training: How the C2000 Configurable Logic Block (CLB) tool integrates custom logic in my design
- ADC Specifications for TMS320F2838xD/S
- ADC Specifications for TMS320F2837xD/S
- ADC Specifications for TMS320F2807x
- ADC Specifications for TMS320F28004x
- C2000™ F2837xD Microcontroller Workshop
- CLA Hands On Workshop
- CLA Usage in Valley Switching Boost Power Factor Correction (PFC) Reference Design
- CLA FAQ on E2E
- C2000 F2837xD Microcontroller 1-Day Workshop Section 1.6: Control Peripherals
## Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

<table>
<thead>
<tr>
<th>Changes from Original (August 2019) to A Revision</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Added new Section 2.2.</td>
<td>8</td>
</tr>
<tr>
<td>• Added new Section 2.3.</td>
<td>10</td>
</tr>
<tr>
<td>• Added new Section 3.1.</td>
<td>16</td>
</tr>
<tr>
<td>• Added new Section 4.1.</td>
<td>24</td>
</tr>
<tr>
<td>• Added new Section 4.2.</td>
<td>26</td>
</tr>
<tr>
<td>• Added new Section 5.1.</td>
<td>35</td>
</tr>
</tbody>
</table>
IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2020, Texas Instruments Incorporated