AN3398

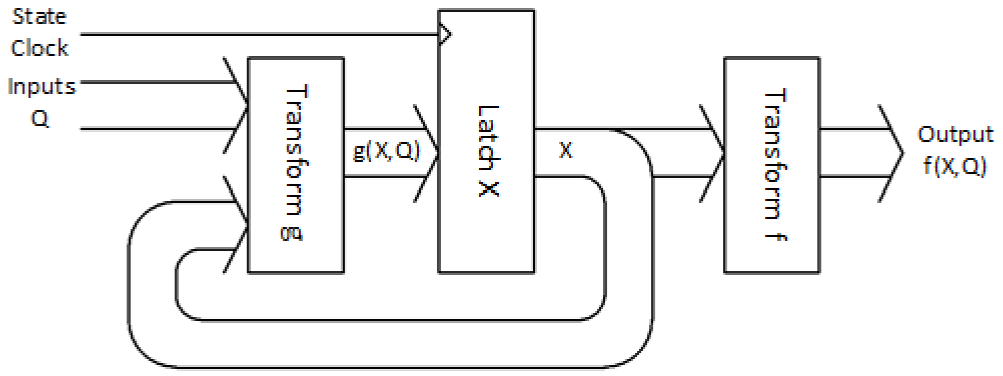
Building Hardware State Machines Using CIPs

Introduction

Author: Keith Curtis, Microchip Technology Inc.

A state machine is a combination of logic, a memory element and feedback. The inputs to the state machine are combined with the current state of the state machine, to determine the next state. The next state becomes the current state when the state clock occurs and the outputs of the state machine are determined by the current state. See Figure 1 for a general block diagram.

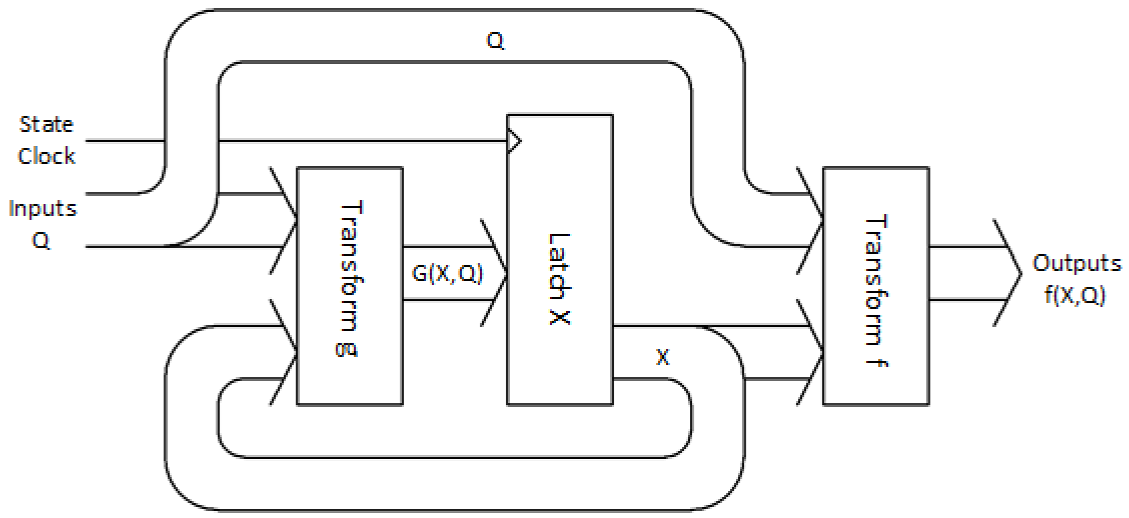
Figure 1. Block Diagram of a Generic State Machine



In the block diagram, input Q is combined with the current output of Latch X. The result, g(X,Q), are the next state and are latched on the active edge of the system clock. The outputs of Latch X can also be passed through additional logic (Transform f) to generate the output. This configuration is typically referred to as a Moore state machine.

An alternative configuration utilizes feed forward from the inputs to the outputs that are not synchronized by the latch. See Figure 2 for a general block diagram.

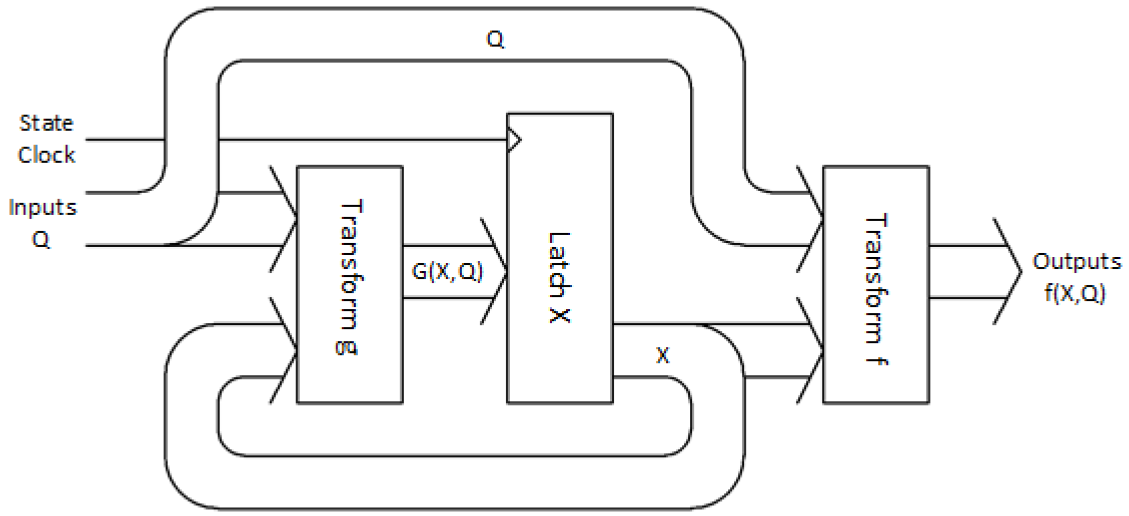
Figure 2. Generic State Machine Block Diagram



In this block diagram, input Q is combined with the current output of Latch X. The result, G(X,Q) are the next states and are latched on the active edge of the system clock. The outputs of Latch X can also be passed through additional logic (Transform f) to generate the output. This configuration is typically referred to as a Moore state machine.

An alternative configuration utilizes feedforward from the inputs to the outputs that are not synchronized by the latch. See Figure 3 for its general block diagram.

Figure 3. Generic State Machine with Feedforward Block Diagram



In this block diagram, the same elements of the Moore design are present, with the addition of the feedforward path from the inputs to the outputs. This configuration is typically referred to as a Mealy state machine.

Typically, a Mealy state machine has fewer states than a Moore design because a single state enables multiple different outputs from the feedforward path. A Moore design is generally considered safer because the outputs are synchronized to the state clock, reducing the possibility of logical race conditions on the state machine output. However, Mealy state machines respond faster to input changes than a Moore state machine.

Table of Contents

1. Overview

This application note will explore several potential designs, which can be implemented with either configuration, using the DMA peripheral.

The DMA peripheral is a byte-wide data transfer peripheral that can be triggered either by software or a selected hardware event. This application will be using the DMA to transfer data from one register to another as:

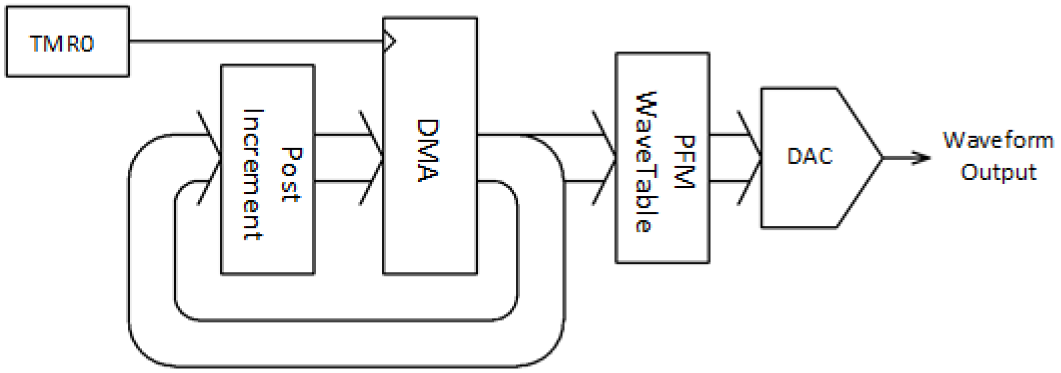
1. The latch element of the state machine, latching and transferring data from the input Transform G to the output Transform F.
2. A generic transfer mechanism to move the state data from the output of the latch to the input of Transform G.
3. A generic transfer mechanism to move the state from the output of the latch to the output Transform F.
4. Or, a generic transfer mechanism to reconfigure a state machine on-the-fly.

While the DMA is the focus of this application note, it will not delve deeply into the configuration of the DMA. For information on the configuration of the DMA, refer to *"Configuring the DMA Peripheral"* Technical Brief (DS90003242) for more information.

2. Moore State Machine (Without an Input Transform G)

To demonstrate a simple Moore configuration, a simple arbitrary waveform generator is presented. The wavetable for the waveform is stored sequentially in Programmable Flash Memory (PFM) and the output is the data register of a Digital-to-Analog Converter (DAC) peripheral. Timer0 provides the periodic state clock, which triggers the DMA peripheral to copy the data from PFM to the DAC on each rising edge. See Figure 2-1 below.

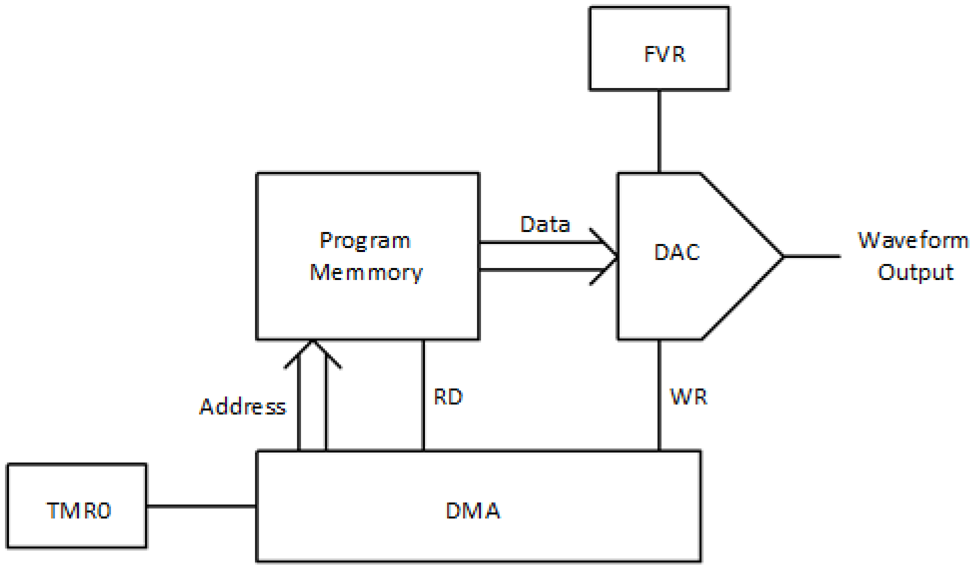
Figure 2-1. Waveform Generator Block Diagram



In this example:

1. TMR0 is configured as the state clock.
2. A table of waveform values is stored in PFM.
3. A 5-bit DAC is used as the state machine output and the FVR is used as the DAC reference.
4. The DMA is configured to read 64 1-byte values from PFM with post-increment and the count register, equal to the size of the PFM table.
5. DMA is further configured to write the 1-byte value to the DAC register using the Fixed Address mode.
6. DMA is configured for continuous operation using TMR0 as the trigger.

Figure 2-2. CIP Connections for a Waveform Generator Block Diagram



In Figure 2-2, Timer0 provides the state clock by triggering the DMA transaction. The transaction loads a value from PFM based on its current source address. The value is then written to the data register of the DAC. Once the transfer is complete, the DMA increments the source address register and reloads the address register if the maximum size of the table is reached. Because the DMA is configured for Continuous mode, this transaction cycle repeats at every state clock transition, out of Timer0, until the DMA or TMR0 is disabled.

While this design doesn't use an external input to the state machine (Transform g), and the feedback is internal to the DMA peripheral (address increment), the basic requirements for a Moore state machine are satisfied. The source address register of the DMA acts as the state machine Latch X. The DMA's post-increment function is the feedback from the output to the input of the latch, providing a sequential source address. The DMA transaction also converts the PFM address to an output value for conversion by the DAC.

3. Moore State Machine (With an Input Transform G)

A LED display driver is implemented as a slight variation on the previous design. In this example, the source memory is GPR memory and the outputs are two contiguous parallel ports, which drive segment and digit drivers. In operation, the DMA will load two values, a segment drive value and a digit enable value. See Table 3-1. These values are transferred to the ports by the DMA, generating the appropriate drive for one of the 7-segment displays.

Table 3-1. Data Storage in GPR Memory

Address	Data
0x00	Digit 1
0x01	`0b00000001`
0x02	Digit 2
0x03	`0b00000010`
0x04	Digit 3
0x05	`0b00000100`
0x06	Digit 4
0x07	`0b00001000`

Because the source is GPR memory, it allows the system code to make changes on-the-fly as the state machine is running, so any values loaded into the GPR memory are automatically displayed on the 7-segment display. See Figure 3-1 for the block diagram and Figure 3-2 for the CIP connections.

Figure 3-1. LED Display Driver Block Diagram

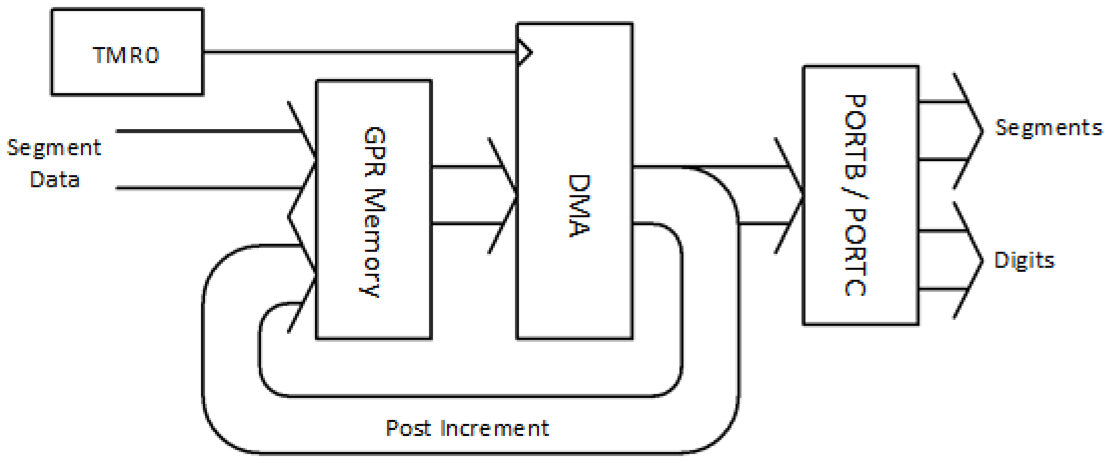
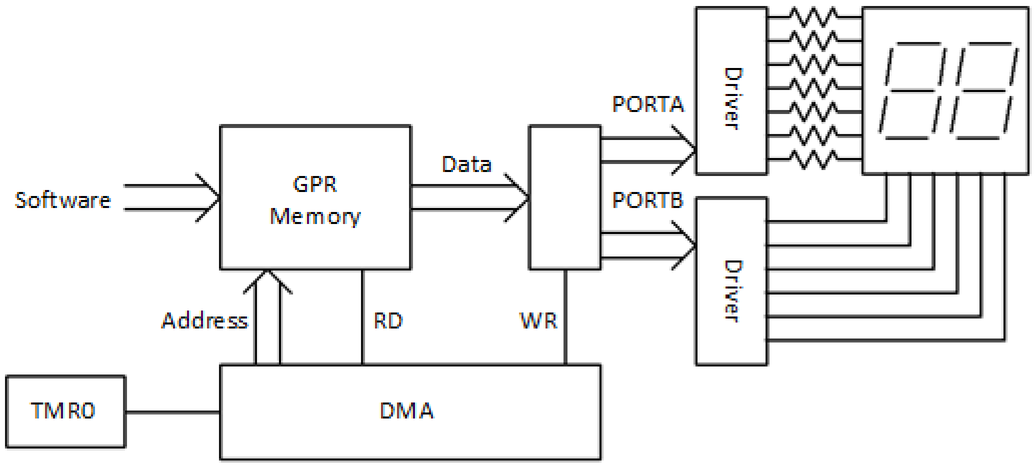


Figure 3-2. CIP Connections for LED Driver



In Figure 3-2, Timer0 provides the state clock by triggering the DMA transaction. The transaction loads the segment and digit values from the GPR memory based on its current source address. The values are then written to PORTB and PORTC. After the transfer of each byte, the DMA increments the source address and reloads the register if the maximum size of the table is reached. Once the value is loaded into the ports, drivers buffer the signals and drive the individual displays. Given the DMA is configured for Continuous mode, the cycle repeats for every state clock transition from Timer0, until the DMA or TMR0 is disabled.

In this design, the input to the state machine is handled by a software update of the display data and the feedback is internal to the DMA peripheral (auto-increment). Together, both systems provide the basic requirements for a Moore state machine. The source address register of the DMA acts as the state machine latch. The DMA's post-increment function is the feedback from the output to the input of the latch, providing a sequential address output. The GPR data, combined with the DMA action, convert the address to data values for both the segment and digit drivers for the displays.

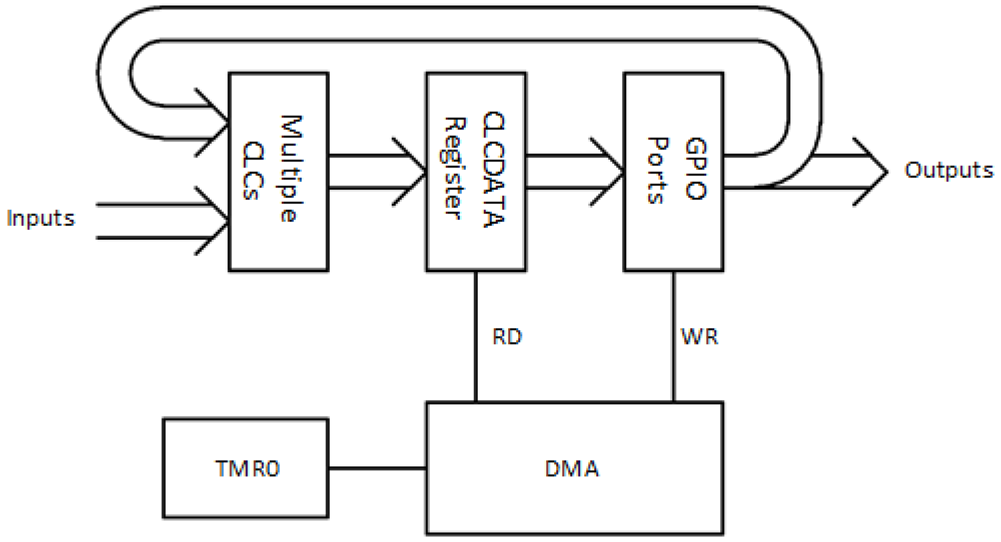
In both of the preceding examples, the DMA peripheral acted as the latch, part of the feedback and a transport mechanism to move the data from memory to the output peripheral. The latch was implemented by the DMA Source Address Pointer (DMAxSPTR). The feedback mechanism was DMAxSPTR's auto-increment, plus the reload countdown in the DMAxSCNT registers. Also, the transport mechanism was the DMA transaction, transferring data pointed at by DMAxSPTR, from memory, to the DAC/port peripherals.

4. Moore State Machine Designs Using CLC and Ports

In application note AN2912, multiple CLC peripherals are combined to create a hardware state machine. One group of the CLCs are used as the state machine latch, while others perform the transforms for the system. While this approach does work, the limited number of CLCs prevents the creation of a complex state machine.

If the CLCs were not needed for the latch function, a somewhat larger, more complex state machine would be possible. One way to accomplish this goal would be to replace the latch function with a combination of the DMA function and a GPIO PORT register. See Figure 4-1.

Figure 4-1. CLC, DMA, PORT Moore State Machine Block Diagram



The CLC peripherals in the design take their input from both the GPIO port, and other system inputs, to form an input transform function using the Boolean functions of the CLC peripheral. The outputs of the CLCs are then latched by the DMA, transferring the CLC outputs to a GPIO port.

Note: The CLC outputs are available as a single byte via the CLCDATA register.

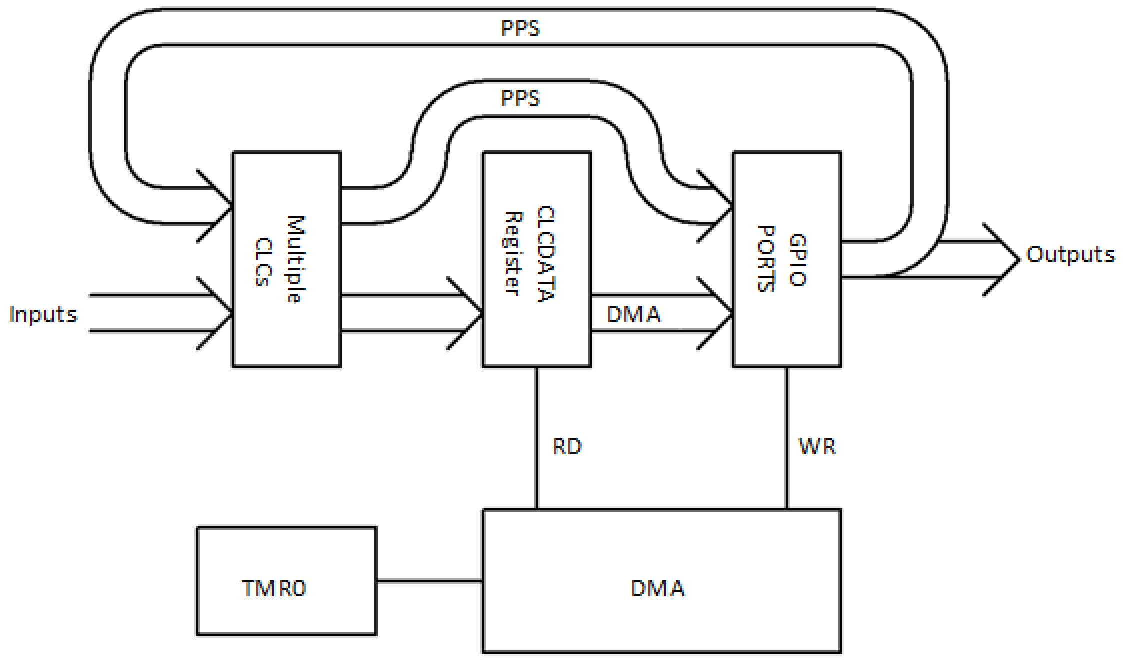
Using the rollover output of TMR0 to trigger the DMA, the outputs of the CLCs are copied from the CLCDATA register and stored in the GPIO port. The individual port pins are then connected to the CLC inputs, using the PPS function. This completes a full latch with feedback from the output to the input.

Note: The full eight bits of the port need not be used for the state machine. Only those port bits configured in PPS to display the LAT data will be affected by the DMA write. Any bits configured to output other CIP functions will continue to output their selected function.

If the microcontroller contains unused CLC peripherals, they can be used to form an output transform Function f. These CLCs would take their inputs from the GPIO port, perform a Boolean transform and then provide their outputs to either other CIP functions, a different GPIO port or the unused GPIO port bits via PPS.

In fact, if the output CLCs also accept inputs from sources other than the state output (GPIO port), then the output transform will include asynchronous inputs, making the design a Mealy state machine. See Figure 4-2.

Figure 4-2. CLC, DMA, PORT Mealy State Machine Block Diagram



5. Using Timer Peripherals as State Machines

For state machine designs that require a minimum number of Active states and long delays between state transitions, a timer can be used as the state machine latch. The advantage of this design is the large number of bits in the latch and the automated increment of the state value.

To provide feedback, the output of the timer is accessible through the CCP and PWM peripherals.

1. If Timer2 is used, then a specific state can be decoded using the PWM logic of the CCP.
2. If Timer1 is used, the decoding can be accomplished by the output compare function in the CCP.
3. If the new 16-bit PWM is used, then left, center, right and variable alignment are available, increasing the options for decoding specific timer values.

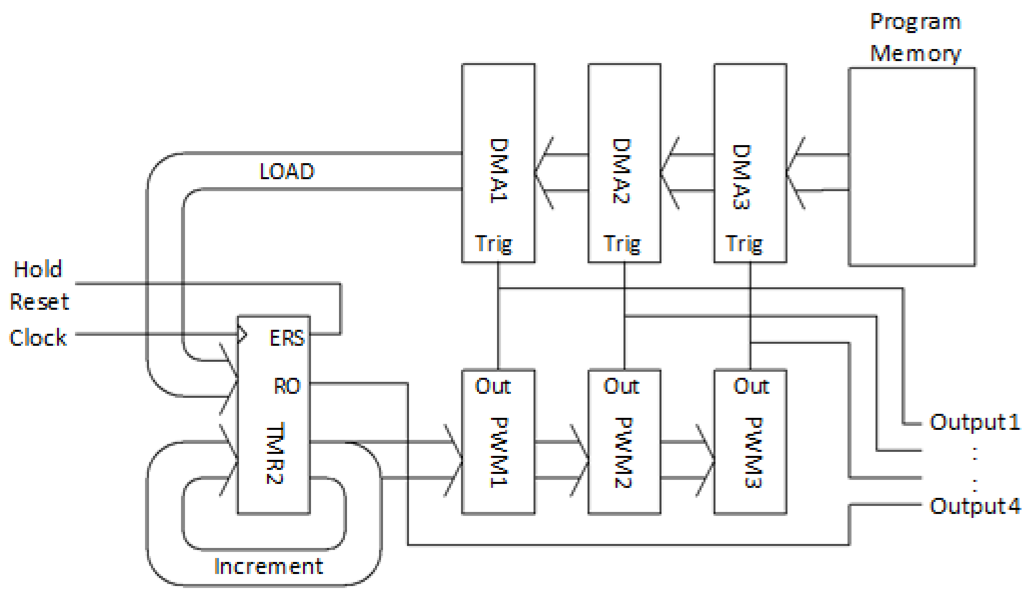
The decoded outputs of the CCP/PWM can then be used to generate outputs, trigger the DMA to change the state value in the timer or reconfigure the timer. See Figure 5-1.

5.1 State Machine Using TMR2 and CCP

Figure 5-1 shows a state machine based on TMR2 and the PWM section of three CCP peripherals. Inputs include the state machine clock and a hold/Reset input to the timer. The output of the timer feeds into CCP1, CCP2 and CCP3. Each PWM output is then routed to the trigger of a DMA module. When the timer out equals the duty cycle setting of the PWM, the associated DMA is triggered and the timer is set to a constant stored in program memory. Each of the PWM outputs also doubles as an output of the state machine, as well as, the rollover output of the timer.

This configuration creates a “Moore State Machine without an Input Transform,” like the first example above. The difference is that this state machine can generate programmable delays between state changes that are both long and variable, under software control. To change the delays, the software need only reprogram the duty cycle register for the appropriate CCP PWM peripherals. In addition, the hold/Reset input allows some control via CIP signals.

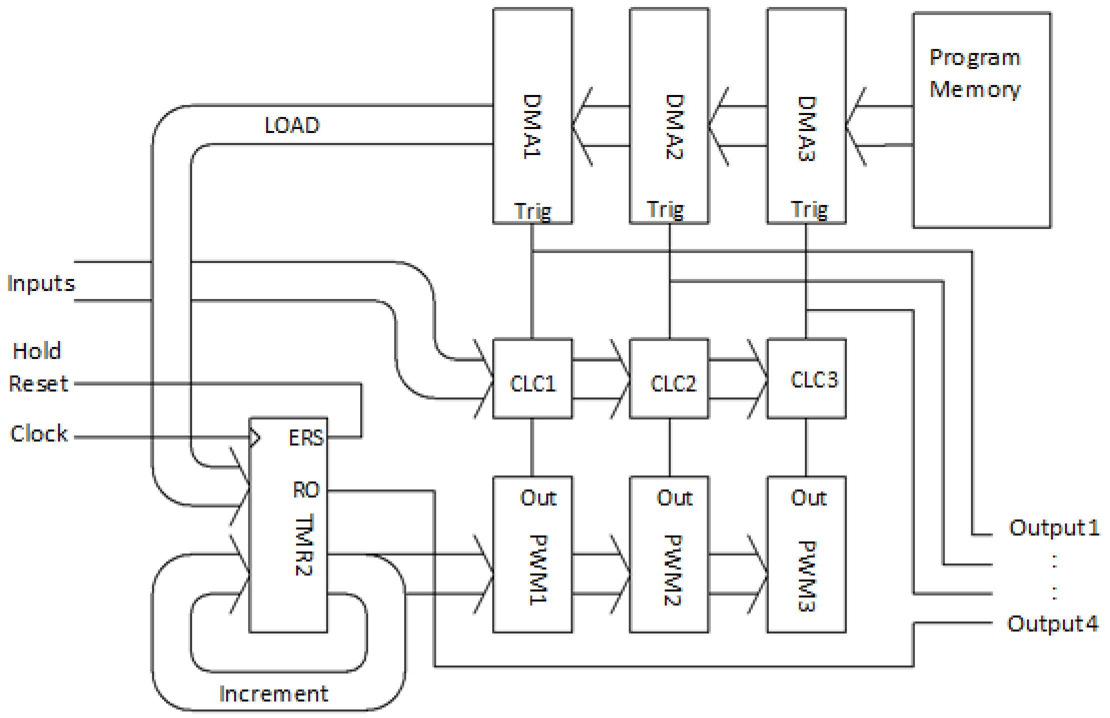
Figure 5-1. State Machine Using TMR2 and CCP



5.2 State Machine using TMR1, CCP and CLC

Figure 5-2 shows a modified version of Figure 5-1 that uses CLC peripherals between the decoded PWM output and the triggers for the DMA. The purpose of the CLC modules is to allow external inputs to the system that combine or enable/disable the feedback based on inputs to the state machine. The CLC combinational logic could also combine different PWM signals to generate more complex state machine outputs.

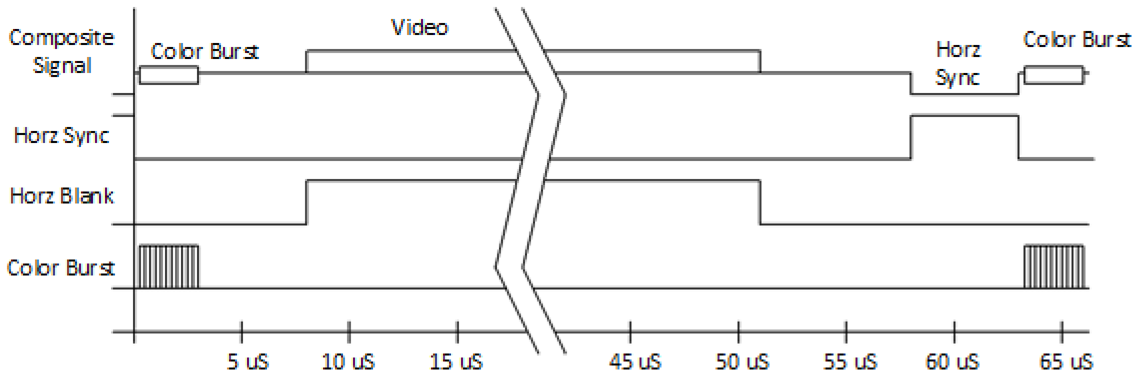
Figure 5-2. State Machine Using TMR1, CCP and CLC



Implementation of a Mealy version of this state machine can also be created by using additional CLC peripherals to combine asynchronous inputs with the PWM outputs, to generate the state machine outputs.

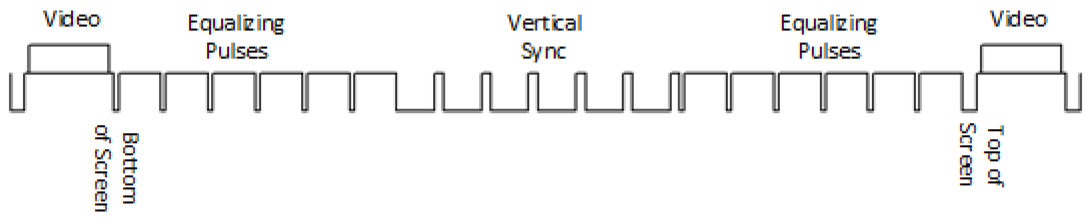
As an example, consider a NTSC horizontal sync timing generator for a television signal. The video pulse chain is 63 μS long with 43 μS of video, 10 μS of blanking and 10 μS of horizontal sync pulse. The horizontal pulse has a 2 μS front porch, a 5 μS pulse and a 3 μS back porch with 2.5 μS of a 3.579545 MHz color burst signal. See Figure 5-3 for a diagram of the timing signals.

Figure 5-3. Horizontal Video Signal Example



During the vertical sync portion of the signal, the 63 μS pulse chain is replaced with a 31.5 μS pulse chain with a 2.3 μS pulse for the first six equalizing pulses, then 27.1 μS vertical sync pulses for another six, and then another six equalizing pulses with the original 2.3 μS, before returning to the normal 63 μS pulse chain. See Figure 5-4 for an example of the vertical timing.

Figure 5-4. Vertical Sync Pulses

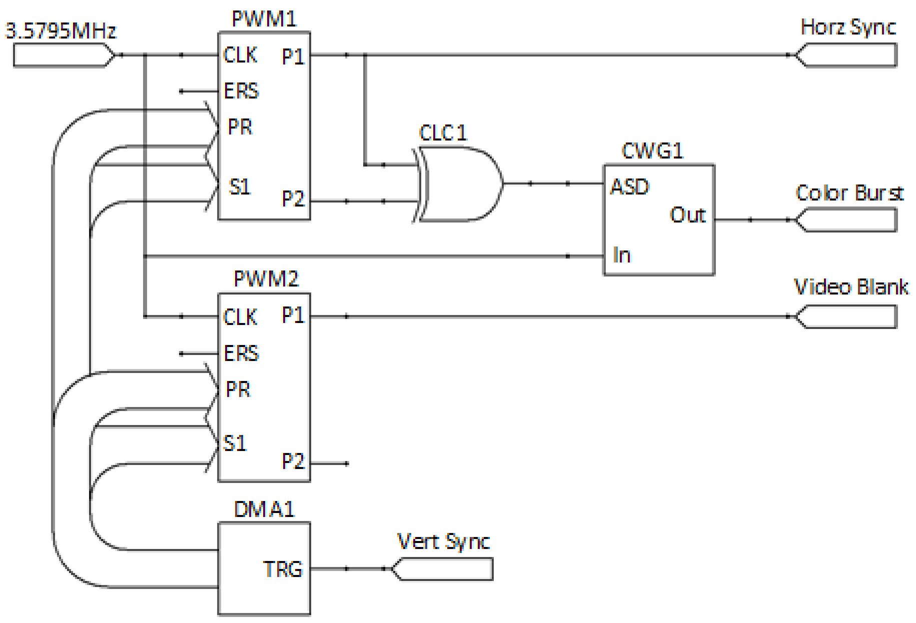


Note: This example will only generate the horizontal timing, not the vertical. However, it does accept inputs from the vertical timing to alter the horizontal timing for the vertical sync portion of the signal. This design will also only implement a non-interlace signal, not a full interlaced timing signal.

5.3 State Machine Using Horizontal Sync Timing

To implement the Horizontal Sync Timing state machine, the new 16-bit PWM module will be used in conjunction with a CLC and DMA peripheral. Figure 5-5 shows the block diagram of the combined CIP peripherals.

Figure 5-5. CIP Block Diagram of the Horizontal Sync Timing System



In the example:

1. PWM1 is configured for dual output, left justified PWM output with a period of 63 µS.
2. P1 of PWM1 is configured to generate the 5 µS horizontal sync pulse.
3. P2 of PWM1 is configured to generate an 8 µS gating signal for the color burst signal.
4. The XOR in CLC1 combines the two signals to generate the 3 µS gating pulse that follows the horizontal sync pulse.
5. The input to the CWG is the raw 3.5795 MHz clock for the color burst.
6. The 3 µS gating pulse drives the auto-shutdown input of the CWG, gating the color burst signal.
7. PWM2 is configured for variable alignment. This allows variable start and stop times for the PWM. This output provides the Video Blank signal.
8. DMA1 is configured to load the period and phase registers for both PWM, from a table in program memory, triggered by the vertical timing chain and the rollover of PWM1's timer.

In normal horizontal video operation, the period of both PWMs is set to 63 µS. PWM1 generates the 5 µS horizontal sync and the 8 µS gating signal. Together, they generate the 3 µS gate for the color burst immediately following the horizontal sync pulse. PWM2 generates the blanking signal, providing 7 μS of blanking before the horizontal sync pulse and 8 μS following the pulse.

The vertical sync system triggers the DMA transfer at the bottom of the page:

1. It reconfigures the horizontal state machine to generate a 2.3 μS equalizing sync pulse with a 31.5 μS period for six pulses. The Blanking PWM is also configured for 100%, to prevent inadvertent video generation.
2. Following the six equalization pulses, the DMA reconfigures the state machine to generate a 27.1 μS sync pulse with a 31.5 μS period for six vertical sync pulses. The blanking PWM remains at 100%.
3. After the vertical sync pulses, the DMA reconfigures the state machine to generate a 2.3 μS sync pulse with a 31.5 μS period for another six equalizing pulses. The Blanking PWM remains at 100%.
4. Finally, it reconfigures the state machine to generate the original horizontal timing and blanking to restart the video at the top of the screen.

Note: To accomplish the reconfiguration using the DMA, the contiguous PWM Mirror registers in the Special Function Register map are used (address = 4020h). This register map is convenient because the period and phase registers have been remapped into adjacent memory locations. See the “Special Function Register Map” figure in the *“PIC18F27/47/57Q43 Data Sheet”* (DS40002147) for more information.

Note: For this design, a 14.31818 MHz (3.5795 x 4) external crystal oscillator was used for the microcontroller clock.

In this example, the state machine latch is the base timer for the two PWM peripherals. The state clock is the 3.5795 MHz clock supplied to the PWM clock inputs. The internal counter increment function provides the feedback and the PWM logic, combined with the external CLC and CWG peripherals, provides the output Transform f. The input to the state machine is provided by the DMA peripheral that reconfigures the two PWMs in response to control signals from the vertical timing state machine and the horizontal sync PWM.

Note: Because PWM1 and PWM2 operate from different internal timers, it is necessary to enable both PWM peripherals at the same time. This is accomplished by using the PWMEN register, which has mirror enable bits for all three PWM modules. Both PWM peripherals were configured and then enabled together by setting the appropriate bits in the register.

6. Conclusions

In this document a variety of different hardware state machine designs have been presented;

1. Using the DMA peripheral to pull sequential state data out of program memory to create a sequential control state machine. This method uses the counter register in the DMA peripheral as the state latch.
2. Using the DMA peripheral to pull sequential state data out of data memory to create sequential control state machines with software-controlled input. This method also uses the counter register in the DMA peripheral as the state latch.
3. Using the DMA peripheral to copy the outputs of the CLC peripherals to a GPIO port, with direction connection of additional CLC outputs to the same port, to create a traditional Mealy hardware state machine. This design also used the latch function in the CLC as the state latch.
4. Finally, a Moore design based on a timer PWM combination that uses the PWM output as an output transform, and the DMA as an input transform that reconfigured the PWM on-the-fly.

While several configurations have been shown, they are not an exhaustive list of the possible configurations. However, the application of a specific example of design will be dictated by the complexity of the design required, and the availability of peripherals on a target device. The examples in this application note were designed using the PIC18F47Q43.

The Microchip Website

Microchip provides online support via our website at http://www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to http://www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: http://www.microchip.com/support

Trademarks

ISBN: 978-1-5224-5673-5

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit http://www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
http://www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880- 3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-72400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-72884388

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820