Application Note

AN2522/D
Rev. 0, 5/2003

DC Motor TPU Function Set (DCm)

By Milan Brejl, Ph.D.

Functional Overview

The DC Motor (DCm) TPU function drives a DC Motor, independently of the CPU. The CPU is required only to set a duty-cycle (*dc*) parameter in the range (–1,1), that determines both the speed and the direction. The function generates unipolar-switched center-aligned PWM signals.

The function set consists of 4 TPU functions:

- DC Motor (DCm)
- Synchronization Signal for DC Motor (DCm_sync)
- Resolver Reference Signal for DC Motor (DCm_res)
- Fault Input for DC Motor (DCm_fault)

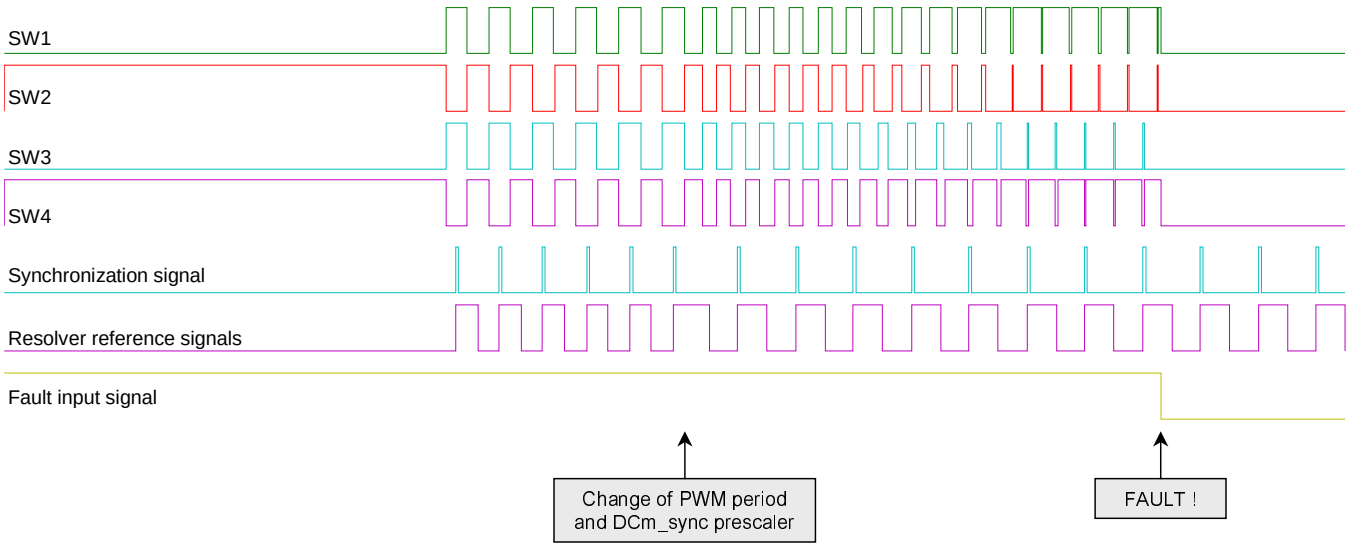

Figure 1. Signals processed by DCm TPU function set

The DCm TPU function generates a 4-channel 2-phase center-aligned PWM signal with dead-time between the top and bottom channels. The Synchronization Signal for the DCm function can be used to generate one or more adjustable signals for a wide range of uses, which are synchronized to the PWM, and track changes in the PWM period. The Resolver Reference Signal for the DCm function can be used to generate one or more 50% duty-cycle adjustable signals that are also synchronized to the PWM.The Fault Input for the DCm function is a TPU input function that sets all PWM outputs low when the input signal goes low. See **Figure 1**.

Function Set Configuration

The DCm function must be used on 4 output channels, and within each phase, the top channel has to be assigned on a lower TPU channel than the bottom channel. One or more channels running Synchronization Signal for DCm as well as Resolver Reference Signals for DCm functions can be added. They can run with different settings on each channel. The function Fault Input for DCm can also be added. It is recommended to use it on channel 15, and to set the hardware option that disables all TPU output pins when the channel 15 input signal is low (DTPU bit = 1). This ensures that the hardware reacts quickly to a pin fault state. Note that it is not only the PWM channels, but all TPU output channels, including the synchronization signals, which are disabled in this configuration.

Table 1 shows the configuration options and restrictions.

Table 1. DCm TPU function set configuration options and restrictions

TPU function	Optional/ Mandatory	How many channels	Assignable channels
DCm	mandatory	4	any 4 channels, SW1 on a lower channel then SW2, SW3 on a lower channel then SW4
DCm_sync	optional	1 or more	any channels
DCm_res	optional	1 or more	any channels
DCm_fault	optional	1	any, recommended is 15 and DTPU bit set

Table 2 shows an example of configuration.

Table 2. Example of configuration

Channel	TPU function	Priority
0	DCm	high
1	DCm	high
2	DCm	high
3	DCm	high
10	DCm_sync	low
11	DCm_res	low
15	DCm_fault	high

Table 3 shows the TPU function code sizes.

Table 3. TPU function code sizes

TPU function	Code size
DCm	108 µ instructions + 8 entries = 116 long words
DCm_sync	26 µ instructions + 8 entries = 34 long words
DCm_res	38 µ instructions + 8 entries = 46 long words
DCm_fault	9 µ instructions + 8 entries = 17 long words

Configuration Order

The CPU configures the TPU as follows.

1. Disables the channels by clearing the two channel priority bits on each channel used (not necessary after reset).
2. Selects the channel functions on all used channels by writing the function numbers to the channel function select bits.
3. Initializes function parameters. The parameters *T*, *DT*, *MPW* and *sync_presc_addr* must be set before initialization. If a DCm_sync channel or a DCm_res channel is used, then its parameters must also be set before initialization.
4. Issues an HSR (Host Service Request) type %10 to one of the DCm channels to initialize all PWM channels. Issues an HSR type %10 to the DCm_sync channels, DCm_res channels and DCm_fault channel, if used.
5. Enables servicing by assigning high, middle or low priority to the channel priority bits. All PWM channels must be assigned the same priority to ensure correct operation. The CPU must ensure that the DCm_sync or DCm_res function is initialized after the initialization of DCm:
   - assign a priority to the PWM channels to enable their initialization
   - if a Synchronization Signal or a Resolver Reference Signal channel is used, wait until the HSR bits are cleared to indicate that initialization of the PWM channels has completed and
   - assign a priority to the DCm_sync or DCm_res channel to enable its initialization

NOTE: *A CPU routine that configures the TPU can be generated automatically using the MPC500_Quick_Start Graphical Configuration Tool.*

Detailed Function Description

DC Motor (DCm)

The DCm TPU function generates a 4-channel, 2-phase unipolar-switched center-aligned PWM signal, with dead-time between the top and bottom channels. In order to charge the bootstrap transistors, the PWM signals start to run 1.6ms after their initialization (at 20MHz TCR1 clock). The functions generate signals corresponding to value 0 of duty-cycle ratio *dc* until the first *dc* value is processed, but for at least one PWM period.

The CPU controls the PWM output by setting the TPU parameters. The duty-cycle ratio *dc* and PWM period *T* can be adjusted during run time. Conversely, dead-time (*DT*) and minimum pulse width (*MPW*) are not supposed to be changed during run time. The duty-cycle ratio *dc* can gain a value in the range (–1, 1). The sign controls the motion system direction, while the absolute value controls the amplitude of the applied voltage.

The following figures show the input *dc* value and corresponding output PWM signals:

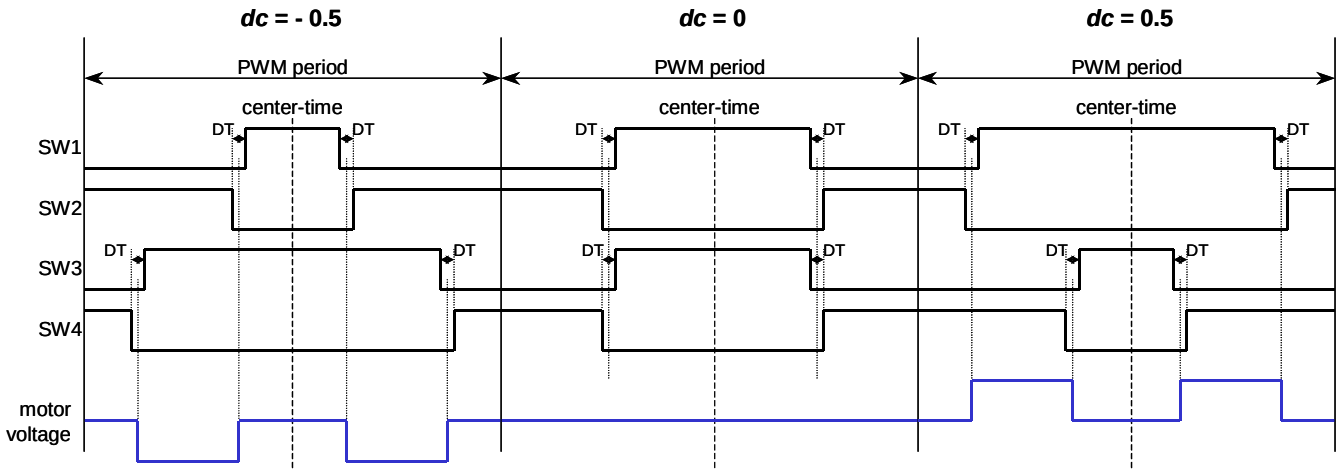

Figure 2. Unipolar switching

The following equations describe how the PWM signal transition times $SW1_{LH}$, $SW2_{LH}$, $SW3_{LH}$, $SW4_{LH}$, $SW1_{HL}$, $SW2_{HL}$, $SW3_{HL}$ and $SW4_{HL}$ are calculated:

$$Tdc_{\alpha} = T \cdot dc$$

$$X = \frac{T + Tdc}{2}$$

$$Y = \frac{T - Tdc}{2}$$

$$A = \frac{X - DT}{2} \qquad C = \frac{Y - DT}{2}$$

$$B = \frac{X + DT}{2} \qquad D = \frac{Y + DT}{2}$$

$$SW1_{LH} = center\_time - A \qquad SW1_{HL} = center\_time + A$$

$$SW2_{HL} = center\_time - B \qquad SW2_{LH} = center\_time + B$$

$$SW3_{LH} = center\_time - C \qquad SW3_{HL} = center\_time + C$$

$$SW4_{HL} = center\_time - D \qquad SW4_{LH} = center\_time + D$$

Host Interface

Written By CPU

Written by both CPU and TPU

Written By TPU

Not Used

Table 4. DCm Control Bits

Name	Options
3 2 1 0 Channel Function Select	DCm function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 – Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Initialization 11 – Stop
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	x – Not used
0 Channel Interrupt Status	x – Not used

Table 5. DCm Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SW1	0	LHtime_1															
	1	HLtime_1															
	2	Tdc															
	3	other_ch_1															
	4	dc															
	5	T															
	6																
	7	fault_pinstate															
SW2	0	LHtime_2															
	1	HLtime_2															
	2	T_copy															
	3	center_time															
	4	DT															
	5	MPW															
	6																
	7																
SW3	0	LHtime_3															
	1	HLtime_3															
	2	L															
	3	other_ch_3															
	4																
	5	sync_presc_addr															
	6																
	7																
SW4	0	LHtime_4															
	1	HLtime_4															
	2	C															
	3	D															
	4																
	5																
	6																
	7																

Table 6. DCm parameter description

Parameter	Format	Description
Parameters written by CPU		
dc	16-bit fractional	duty-cycle ratio in the range <–1,1)
T	16-bit unsigned integer	PWM period in number of TCR1 TPU cycles
DT	16-bit unsigned integer	Dead-time in number of TCR1 TPU cycles
MPW	16-bit unsigned integer	Minimum pulse width in number of TCR1 TPU cycles. See **Performance** for details.
sync_presc_addr	8-bit unsigned integer	address of synchronization channel *prescaler* parameter: $X4, where X is synchronization channel number. $0 if no synchronization channel is used.
Parameters written by TPU		
fault_pinstate	0 or 1	If fault channel is used, state of fault pin: 0 ... low 1 ... high
Other parameters are just for TPU function inner use.		

Performance

Table 7. DCm State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
INIT	80	24
STOP	26	0
C10	6	1
C1	80	13
C20	6	2
C2	48	19
LH	2	1
HL	2	1

Execution times do not include the time slot transition time (TST = 10 or 14 IMB clocks)

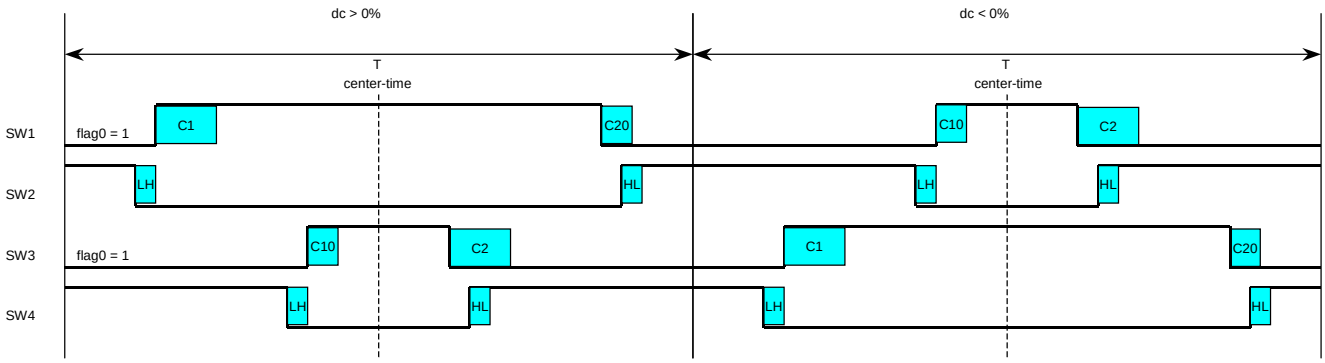

Figure 3. DCm timing

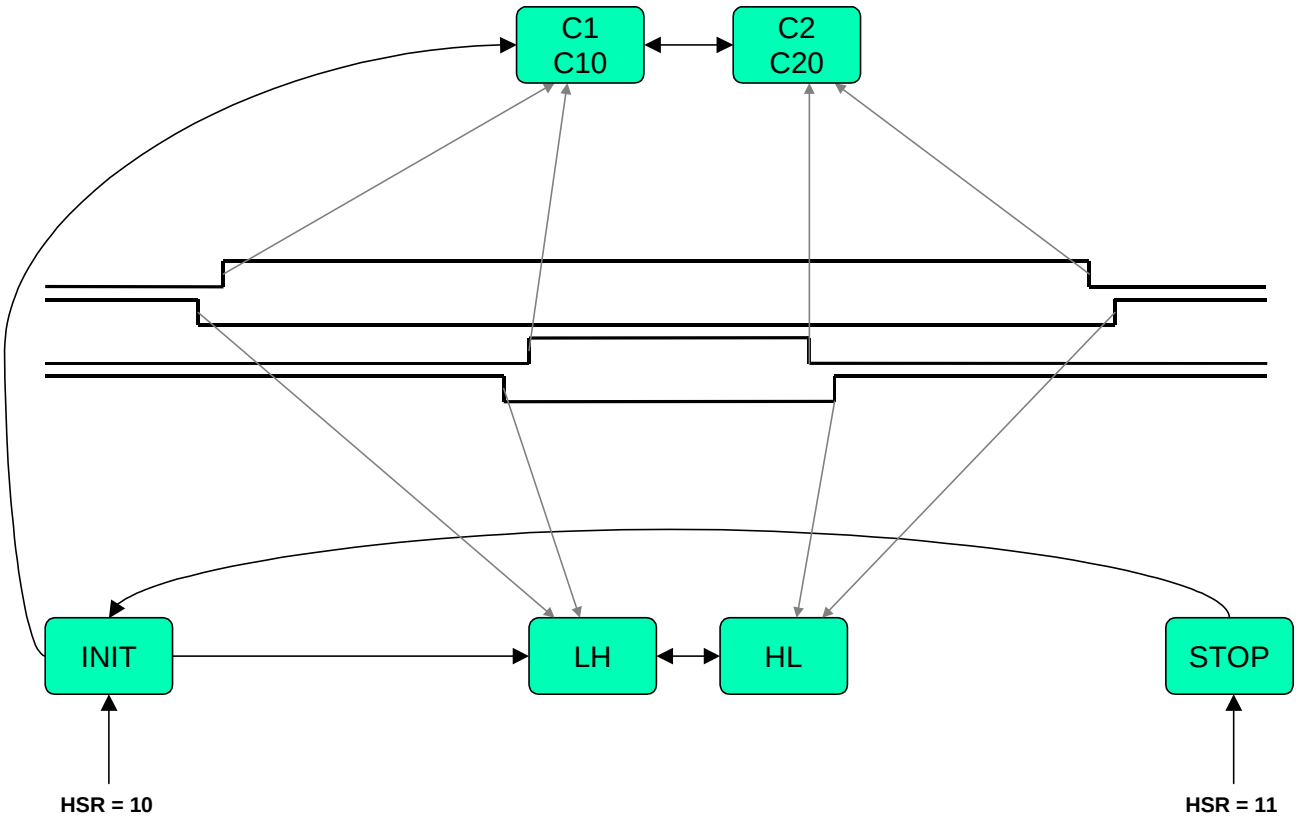

Figure 4. DCm state diagram

Minimum Pulse Width

The TPU cannot generate PWM signals with duty cycle ratios very close to 0% or 100%. This is the case when the dc value is close to 1 or –1. The minimum pulse width that the TPU can be guaranteed to correctly generate is determined by the TPU function itself and by the activity on the other channels. When the TPU function is requested to generate a narrower pulse a collision can occur. To prevent this, the parameter *MPW* (minimum pulse width) is introduced. The TPU function DCm limits the narrowest generated pulse widths to *MPW*. The CPU program should check the maximum absolute value of *dc* to prevent this limitation, or take into account the non-linear performance when *dc* moves towards the boundary values and the limitation is reached by the TPU. The maximum absolute value of *dc* should satisfy:

$$|dc| \leq 1 - \frac{2(MPW + DT)}{T}$$

The *MPW* is written by the CPU. The *MPW* depends on the whole TPU unit configuration, especially the lengths of longest states of the other functions running on the same TPU and their priorities. The *MPW* has to be correctly calculated at the time of the complete TPU unit configuration.

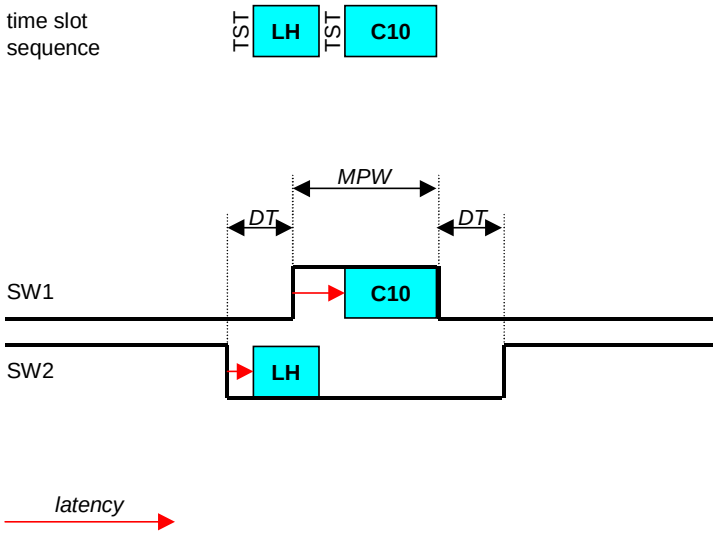

Figure 5. Worst case timing – case one

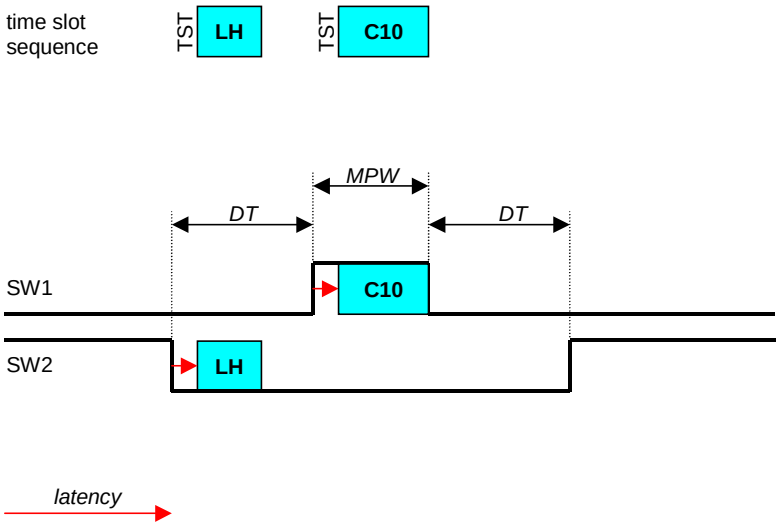

Figure 6. Worst case timing – case two

The minimum pulse width can be calculated according to **Figure 5** or **Figure 6**. These illustrates two possible worst cases for timing in the case when only the DCm function is running on one TPU.

According to the **Figure 5** the *MPW* is 28 IMB clock cycles – *DT*. According to **Figure 6** the *MPW* is 16 IMB clock cycles. In summary the MPW parameter value is equal to 28 IMB clock cycles – *DT*, with a minimum value of at least 16 IMB clock cycles.

Note that the *MPW*, as well as the *DT*, are entered into the parameter RAM in TCR1 clock cycles rather that IMB clock cycles. It is recommended for the DCm function, to configure the TCR1 clock to its maximum speed, which is the IMB clock divided by 2. In this case the *MPW* = 14 – *DT*, with a minimum value of 8.

When other functions are running concurrently on the same TPU, the longest state of each function with its time-slot transition can increase the calculated *MPW* value. The DCm_fault function does not affect the *MPW*. The DCm_sync, if used, increase the *MPW* value by 22 (44 IMB clock cycles). The DCm_res, if used, increase the *MPW* value by 20 (40 IMB clock cycles).

If a lower value than the one calculated is assigned to the *MPW* parameter, the motion system can run with a higher motor voltage amplitude, but with a very low probability risk, that the dead-time is not kept.

You can also use the Worst-Case Latency (WCL) that is automatically calculated by the MPC500_Quick_Start Graphical Configuration Tool. It can serve as a good approximation of *MPW*. The calculated WCL is always longer than the real-case is. Let the WCL be calculated after the configuration of the

TPU channels and then find the longest WCL value within all DCm PWM channels. Convert the number, from IMB clock cycles to TCR1 clock cycles, to get the *MPW*.

Synchronization signal for DC Motor (DCm_sync)

The DCm_sync TPU function uses information obtained from DCm PWM functions, the actual PWM center times and the PWM periods. This allows a signal to be generated, that tracks the changes in the PWM period and is always synchronized with the PWM. The synchronization signal is a positive pulse generated repeatedly after the *prescaler* or *presc_copy* PWM periods (see next paragraph). The low to high transition of the pulse can be adjusted by a parameter, either negative or positive, to go before or after the PWM period center time of a number of TCR1 TPU cycles. The pulse width *pw* is another synchronization signal parameter.

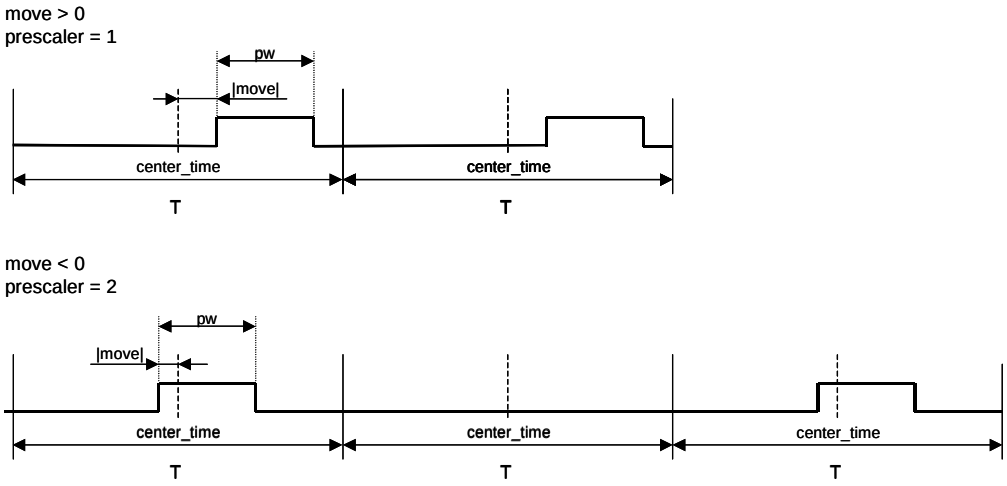

Figure 7. Synchronization signal adjustment examples

Synchronized Change of PWM Prescaler And Synchronization Signal Prescaler

The DCm_sync TPU function actually uses the *presc_copy* parameter instead of the *prescaler* parameter. The *prescaler* parameter holds the prescaler value that is copied to the *presc_copy* by the DCm_bottom function at the time of the PWM parameters reload. This ensures that new prescaler values for the PWM signals, as well as the synchronization signal, are applied at the same time. Write the synchronization signals *prescaler* parameter address to the *sync_presc_addr* parameter to enable this mechanism. Write 0 to disable it, and remember to set the synchronization signal *presc_copy* parameter instead of the *prescaler* parameter in this case.

Host Interface

Written By CPU

Written by both CPU and TPU

Written By TPU

Not Used

Table 8. DCm_sync Control Bits

Name	Options
3 2 1 0 Channel Function Select	DCm_sync function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 – Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Initialization 11 – Not used
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	0 – Channel Interrupt Disabled 1 – Channel Interrupt Enabled
0 Channel Interrupt Status	0 – Interrupt Not Asserted 1 – Interrupt Asserted

TPU function DCm_sync generates an interrupt after each low to high transition.

Table 9. DCm_sync Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Synchronization channel	0	move															
	1	pw															
	2	prescaler															
	3	presc_copy															
	4	time															
	5	dec															
	6	T_copy															
	7																

Table 10. DCm_sync parameter description

Parameter	Format	Description
Parameters written by CPU		
move	16-bit signed integer	The number of TCR1 TPU cycles to forego (negative) or come after (positive) the PWM period center time
pw	16-bit unsigned integer	Synchronization pulse width in number of TCR1 TPU cycles.
prescaler	16-bit unsigned integer	The number of PWM periods per synchronization pulse – use in case of synchronized prescalers change
presc_copy	16-bit unsigned integer	The number of PWM periods per synchronization pulse – use in case of asynchronized prescalers change
Parameters written by TPU		
Other parameters are just for TPU function inner use.		

Performance

There is one limitation. The absolute value of parameter *move* has to be less then a quarter of the PWM period *T*.

$$|move| < \frac{T}{4}$$

Table 11. DCm_sync State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
INIT	12	5
S1	12	6
S2	8	3
S3	16	7

NOTE: *Execution times do not include the time slot transition time (TST = 10 or 14 IMB clocks)*

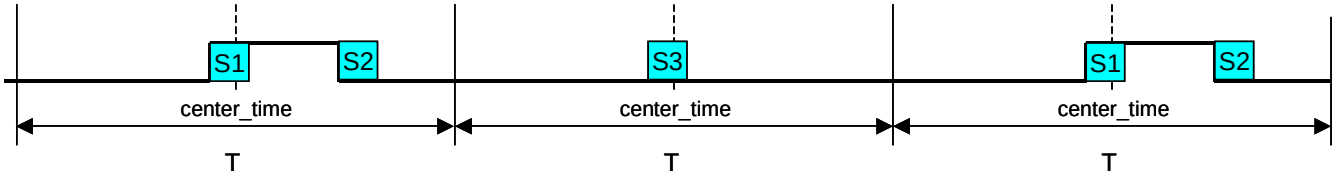

Figure 8. DCm_sync timing

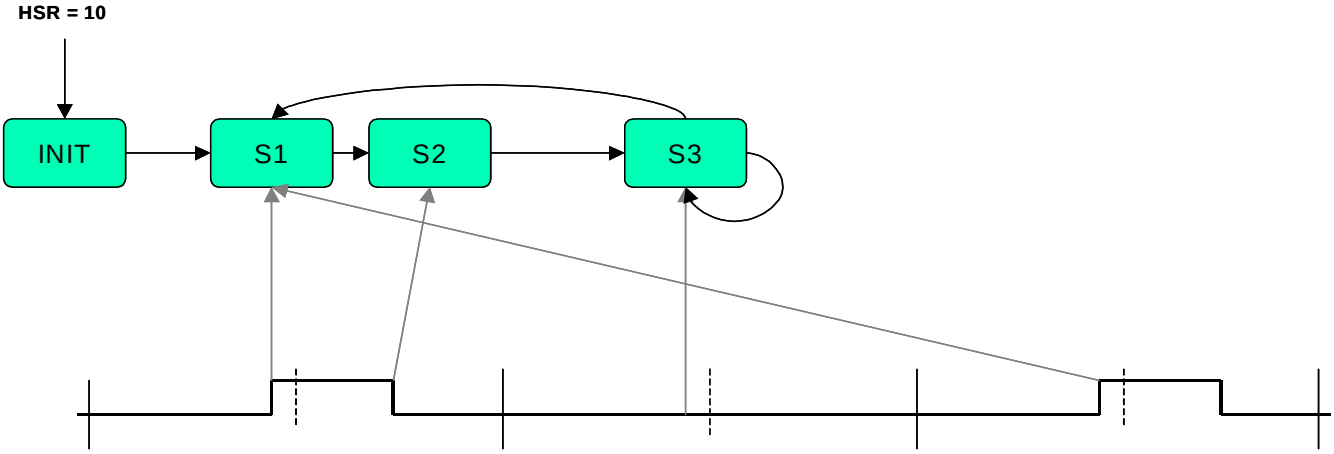

Figure 9. DCm_sync state diagram

Resolver Reference Signal for DC Motor (DCm_res)

The DCm_res TPU function uses information read from the DCm PWM functions, the actual PWM center times and the PWM periods. This allows a signal to be generated, which tracks the changes of the PWM period and is always synchronized with the PWM. The resolver reference signal is a 50% duty-cycle signal with a period equal to *prescaler* or synchronization channel *presc_copy* PWM periods (see next paragraph). The low to high transition of the pulse can be adjusted by a parameter, either negative or positive, to go before or after the PWM period center time of a number of TCR1 TPU cycles.

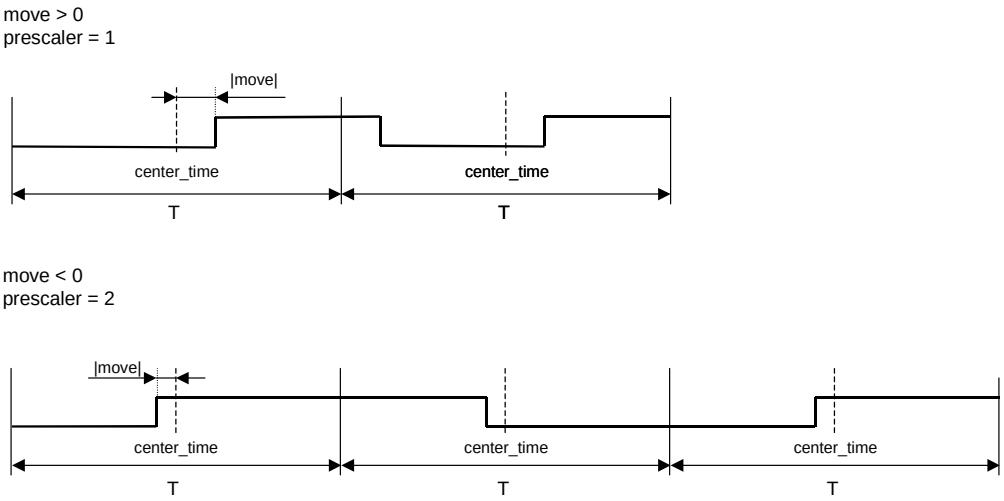

Figure 10. Resolver reference signal adjustment examples

Synchronized Change of PWM Prescaler And Resolver Reference Signals Prescaler

The DCm_res TPU function can inherit the Synchronization Signal prescaler that is synchronously changed with PWM prescaler. Write the synchronization signals *presc_copy* parameter address to the *presc_addr* parameter to enable this mechanism. Write 0 to disable it, and in this case set *prescaler* parameter to directly specify prescaler value.

Host Interface

Written By CPU

Written by both CPU and TPU

Written By TPU

Not Used

Table 12. DCm_res Control Bits

Name	Options
3 2 1 0 Channel Function Select	DCm_res function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 – Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Initialization 11 – Not used
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	x – Not used
0 Channel Interrupt Status	x – Not used

Table 13. DCm_res Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Resolver	0	move															
	1																
	2	presc_addr															
	3	prescaler															
	4	time															
	5	dec															
	6	T_copy															
	7																

Table 14. DCm_res parameter description

Parameter	Format	Description
Parameters written by CPU		
move	16-bit signed integer	The number of TCR1 TPU cycles to forego (negative) or come after (positive) the PWM period center time
presc_addr	16-bit unsigned integer	$00X6, where X is a number of Synchronization Signal channel, to inherit Sync. channel prescaler or $0000 to enable direct specification of prescaler value in prescaler parameter
prescaler	1, 2, 4, 6, 8, 10, 12, 14, ...	The number of PWM periods per synchronization pulse – use when apresc_addr = 0
Parameters written by TPU		
Other parameters are just for TPU function inner use.		

Performance

There is one limitation. The absolute value of parameter *move* has to be less than a quarter of the PWM period *T*.

$$|move| < \frac{T}{4}$$

Table 15. DCm_res State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
INIT	12	5
S1	26	9
S3	16	7

NOTE: *Execution times do not include the time slot transition time (TST = 10 or 14 IMB clocks)*

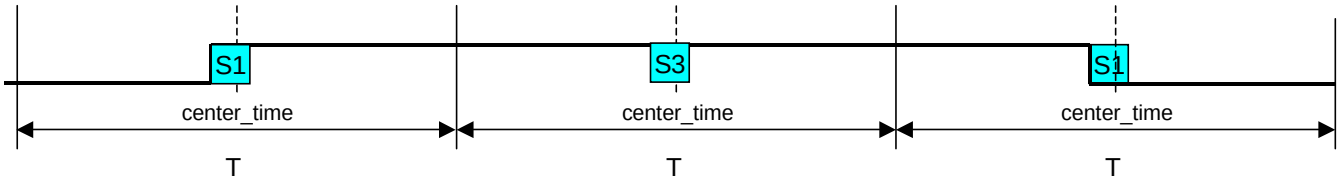

Figure 11. DCm_res timing

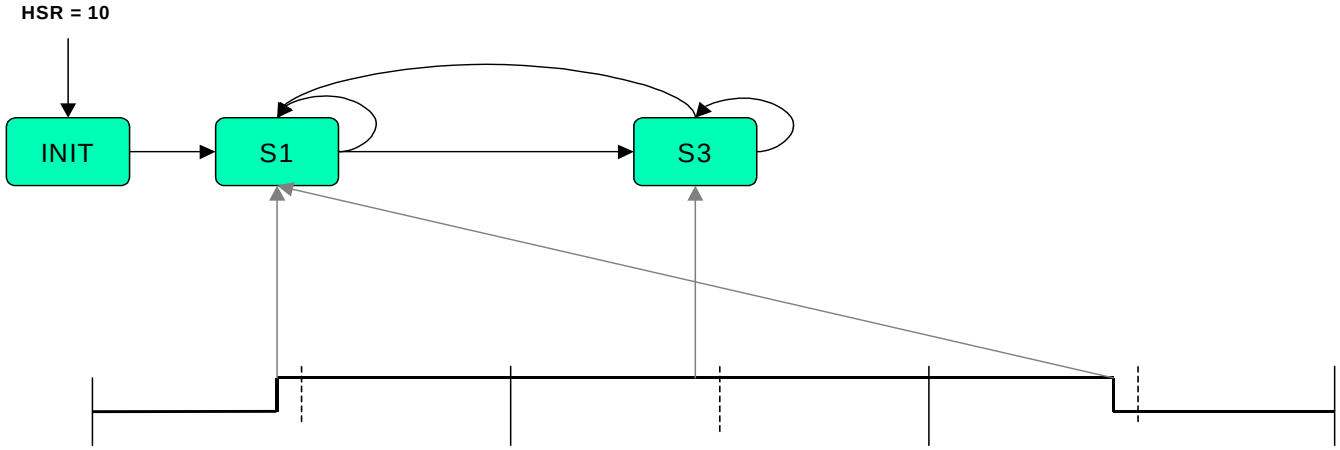

Figure 12. DCm_res state diagram

Fault Input for DC Motor (DCm_fault)

The DCm_fault is an input TPU function that monitors the pin, and if a high to low transition occurs, immediately sets all PWM channels low and cancels all further transitions on them. The PWM channels, as well as the synchronization and resolver reference signal channels (if used), have to be initialized again to start them running.

The function returns the actual pinstate as a value of 0 (low) or 1 (high) in the parameter *fault_pinstate*. The parameter is placed on the SW1 channel to keep the fault channel parameter space free.

Host Interface

Written By CPU

Written by both CPU and TPU

Written By TPU

Not Used

Table 16. DCm_fault Control Bits

Name	Options
3 2 1 0 Channel Function Select	DCm_fault function number (Assigned during assembly the DPTRAM code from library TPU functions)
1 0 Channel Priority	00 – Channel Disabled 01 – Low Priority 10 – Middle Priority 11 – High Priority
1 0 Host Service Bits (HSR)	00 – No Host Service Request 01 – Not used 10 – Initialization 11 – Not used
1 0 Host Sequence Bits (HSQ)	xx – Not used
0 Channel Interrupt Enable	0 – Channel Interrupt Disabled 1 – Channel Interrupt Enabled
0 Channel Interrupt Status	0 – Interrupt Not Asserted 1 – Interrupt Asserted

TPU function DCm_fault generates an interrupt when a high to low transition appears.

Table 17. DCm_fault Parameter RAM

Channel	Parameter	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Fault input	0																
	1																
	2																
	3																
	4																
	5																
	6																
	7																

Table 18. DCm_fault parameter description

Parameter	Format	Description
Parameters written by TPU		
fault_pinstate	0 or 1	State of fault pin: 0 ... low 1 ... high

Performance

Table 19. DCm_fault State Statistics

State	Max IMB Clock Cycles	RAM Accesses by TPU
INIT	8	2
FAULT	32	1
NO_FAULT	4	1

NOTE: *Execution times do not include the time slot transition time (TST = 10 or 14 IMB clocks)*

Figure 13. DCm_fault timing

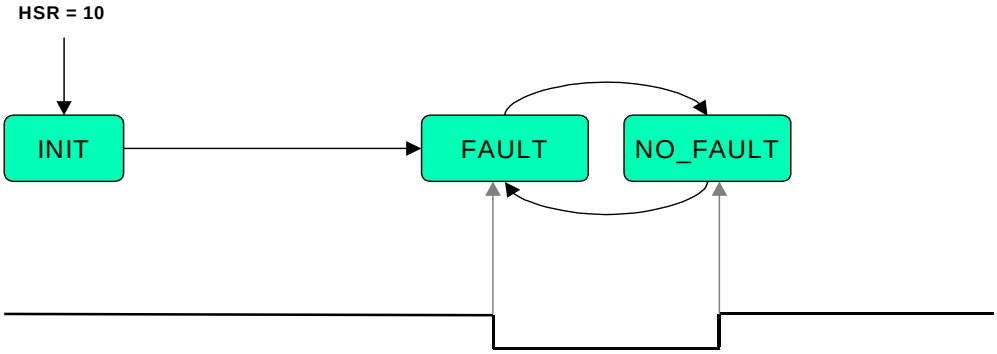

Figure 14. DCm_fault state diagram

[redacted]

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

freescale™
semiconductor

AN2522/D
Rev. 0
5/2003