DSPSM1602

Intelligent Movement and Feedback Controller

for PSM Series Motorized Slide Potentiometer

FEATURES

- Fully integrated motor-pot controller
- Touch detection and action
- Position recall functionality
- Haptic feedback support
- Programmable force feedback modes
- Programmable tactile feedback events
- Motor thermal profiling and protection
- 3.0 to 5.5V logic voltage range
- 4.5 to 16V motor voltage range
- QWIIC compatible

APPLICATIONS

- Re-configurable input devices
- Haptic feedback for robots
- Educational systems
- Audio mixing consoles
- Broadcast mixing consoles

DESCRIPTION

DSPSM1602 is an intelligent controller for motorized potentiometers that will attach as a shield to a Bourns PSM01 or PSM60 motorized potentiometer and provides the functionality necessary to read and drive the potentiometer from any I^2C capable microcontroller.

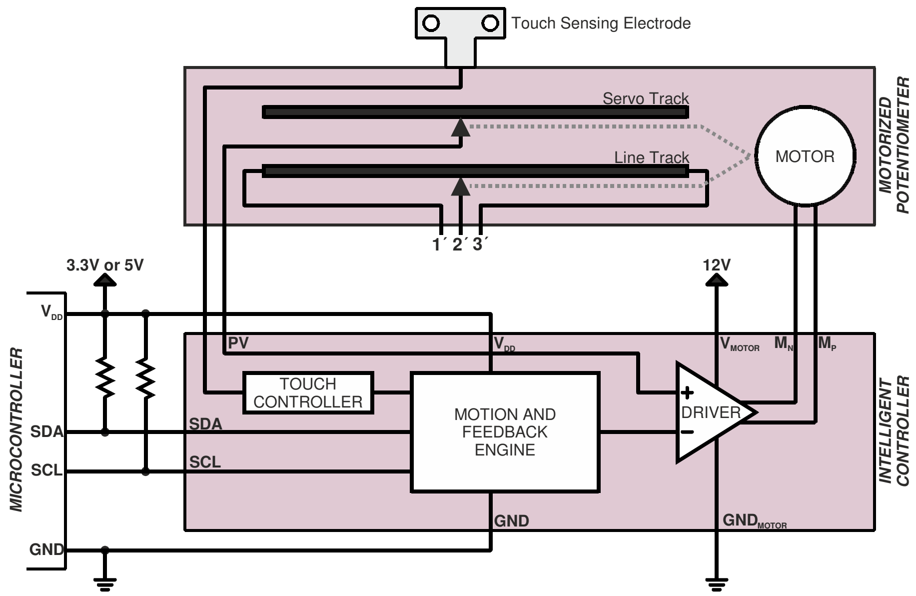
This controller will allow digital read-out of the potentiometer setting as well as motorized recall of positions.

In addition it is possible to trigger tactile feedback events such as vibration as well as configuring force feedback modes such as a center detent, or multiple detents for enumerated inputs.

It is also possible to implement haptic feedback for robotics applications, enabling the user to “feel” the forces applied on the robotic actuator.

The controller contains a touch detector, motion profile generator, PID controller, motor driver and logic, providing a fast and easy path from specification to implementation.

TYPICAL APPLICATION



ABSOLUTE MAXIMUM RATINGS

Supply Voltages
- **V_{DD}** to **GND**... 5.5 V
- **V_{MOTOR}** to **GND_{MOTOR}**................................... 16 V

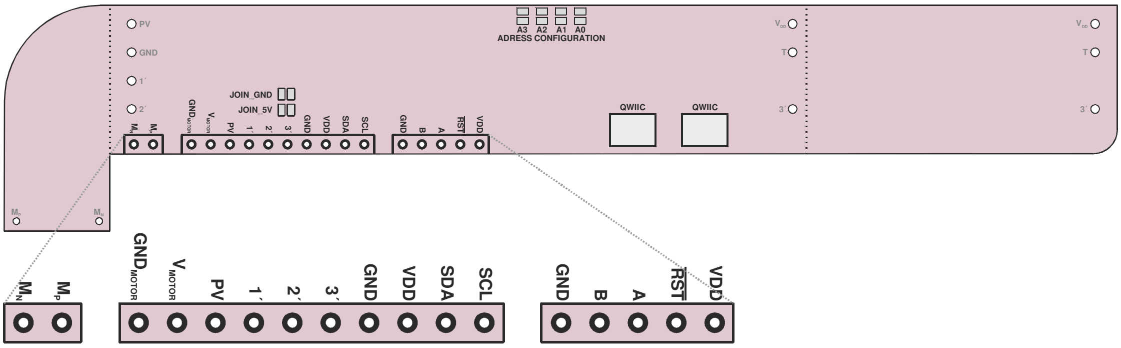
Ground Potential
- **GND** to **GND_{MOTOR}**.............................. ±100 mV

Motor Current (Continuous).................. 1000 mA

Temperature Range...................... -40°C to 85°C

Storage Temperature Range...... -65°C to 125°C

PIN CONFIGURATION



V_{DD}	Digital Supply Voltage. Typically 3.3V or 5.0V.
GND	System Ground. Typically tied to **GND_{MOTOR}**.
V_{MOTOR}	Motor Supply Voltage. Typically 12V
GND_{MOTOR}	Motor Ground. Typically tied to **GND**.
SCL	Connect to I²C master SCL, requires external pull-up on the I²C bus.
SDA	Connect to I²C master SDA, requires external pull-up on the I²C bus.
PV	Connected to *Servo Track wiper*. Typically not connected, but may be connected to a high impedance input, for example an operational amplifier or an ADC.

1´	Connected to *Line Track* near end-point. Uncommitted, can be used freely by external circuitry.
2´	Connected to *Line Track* wiper. Uncommitted, can be used freely by external circuitry.
3´	Connected to *Line Track* far end-point. Uncommitted, can be used freely by external circuitry.
$\overline{RST}$	Active low reset input for the motion and feedback engine. Has an on-board pull-up resistor and is typically not connected.
M_P, M_N	Motor outputs. Typically not connected.
A, B	Used for firmware programming and reserved for future application. Do not connect.

Note: The headers marked **QWIIC** will each accept one JST 4-pin connector for easy prototyping and daisy-chaining. When connected to a QWIIC-enabled microcontroller the only other connections required are **V_{MOTOR}** and **GND_{MOTOR}**. Please look online for more information on the QWIIC connect system.

ELECTRICAL SPECIFICATION

Parameter	Min	Typ	Max	Unit	Conditions
DIGITAL SUPPLY					
Supply Voltage ($\mathbf{V_{DD}}$)	3.0	3.3/5.0	5.5	V	
Supply Current ($\mathbf{I_{DD}}$)		16mA	50	mA	
MOTOR SUPPLY					
Motor Voltage ($\mathbf{V_{MOTOR}}$)	4.5	12	16	V	
Motor Current ($\mathbf{I_{MOTOR}}$)			500	mA	
Motor Standby Current		0	1	mA	Motor off
LOGIC LEVELS					
Logic High Output ($\mathbf{V_{OH}}$)	$\mathbf{V_{DD}}$ – 0.8			V	
Logic Low Output ($\mathbf{V_{OL}}$)			0.8	V	
Logic High Input ($\mathbf{V_{IH}}$)	0.7 × $\mathbf{V_{DD}}$			V	
Logic Low Input ($\mathbf{V_{IL}}$)			0.30 × $\mathbf{V_{DD}}$	V	
ANALOG					
PV impedance	0		10	kΩ	10kΩ servo track, position dependent

PERFORMANCE SPECIFICATION

Parameter	Min	Typ	Max	Unit	Conditions
Digital Readout Resolution			12	bits	
Digital Readout SNR		1		bits	**PV_FILTER** >= 3, motor standby
Movement Command Error		0.5	2	% full scale	$\mathbf{V_{MOTOR}}$ = 12V, factory configuration
Full Range Movement		250	300	ms	
PWM frequency		100		kHz	
Control loop bandwidth		25		kHz	
PID controller bandwidth		1		kHz	
Motion controller bandwidth		250		Hz	

CONFIGURATION JUMPERS

ADRESS SELECTION (A0...A3)

The I^2C slave address of the device is selected by configuring solder jumpers ***A0***, ***A1***, ***A2*** and ***A3***.

The pads can be soldered together, but will also accept a 0603 component such as a 0-ohm resistor.

A3 is used to select an address range and will in addition to being left open or being bridged, also accept a 5% resistor for extended addressing options.

A2-***A0*** are used to select an address within that range.

The following tables describe the configuration of ***A0***-***A3***:

A3	Range
open	*16*
bridged	*24*
6.8k	*32*
15k	*40*
27k	*48*
47k	*56*
82k	*64*
150k	*72*
330k	*80*

A2	A1	A0	Address
open	*open*	*open*	*0*
open	*open*	*bridged*	*1*
open	*bridged*	*open*	*2*
open	*bridged*	*bridged*	*3*
bridged	*open*	*open*	*4*
bridged	*open*	*bridged*	*5*
bridged	*bridged*	*open*	*6*
bridged	*bridged*	*bridged*	*7*

The I^2C slave address is the sum of *Range* and *Address*.

Example: ***A3*** *is loaded with 27k,* ***A2*** *and* ***A1*** *are left open,* ***A0*** *is bridged. The configured* I^2C *slave address is 48+1 = 49.*

By default, all jumpers are open and the **default address is 16.**

GROUND SEPARATION (JOIN_GND)

Joins motor ground (***GND_{MOTOR}***) and digital ground (***GND***) together.

This solder jumper is **bridged by default**.

For applications where it is necessary to separate motor ground from digital ground (e.g. to reduce interference) it is possible to cut this jumper.

GND_{MOTOR} and ***GND*** must still be tied together, but by cutting this jumper is is possible to externally tie them together using, for example, a ferrite.

For applications that require low-side current sensing for the motor, it is possible to use a small external resistor between ***GND_{MOTOR}*** and system ground.

The pads can be soldered together, but will also accept a 0603 component such as a ferrite or a 0-ohm resistor.

POWER SEPARATION (JOIN_5V)

This solder jumper is **open/not bridged by default.**

For testing, prototyping or specific applications is may be useful to tie digital power (***V_{DD}***) together with motor power (***V_{MOTOR}***).

This allows the entire device (controller & motor) to run from a single power connection (e.g. a single QWIIC connector [Note1]), although the motor will be operating at reduced strength.

This configuration is only supported when *4.5V <* ***V_{DD}*** *< 5.5V.*

The pads can be soldered together, but will also accept a 0603 component such as a ferrite or a 0-ohm resistor.

Note1 *Since QWIIC is 3.3V only, this configuration does not adhere to the QWIIC standard. For full QWIIC compatibility, external power must be supplied to* ***V_{MOTOR}***

DS PROTOTYP

ASSEMBLY INSTRUCTIONS

BOARD AND POTENTIOMETER

If you've purchased the controller and potentiometer separately, you will need to install the potentiometer onto the circuit board.

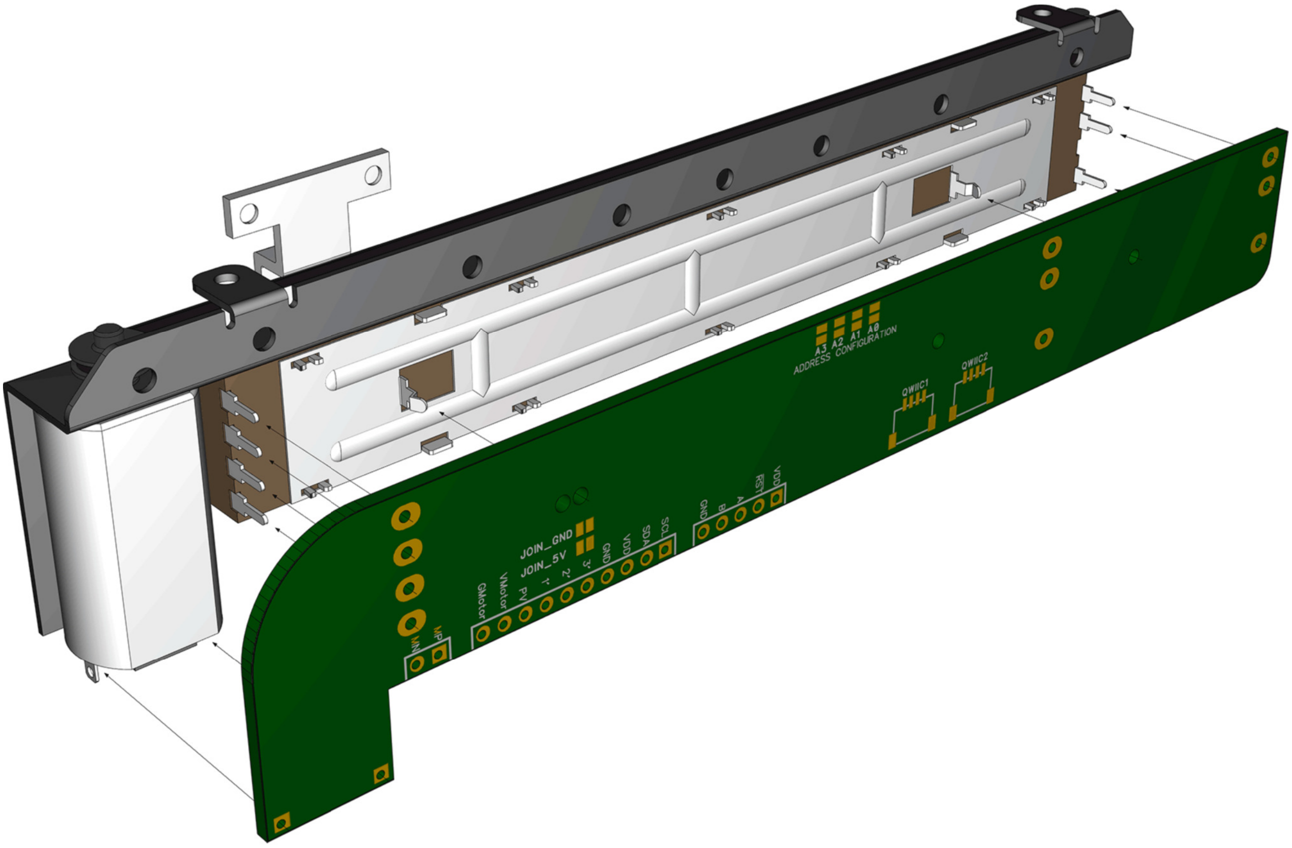
Carefully insert the potentiometer into the circuit board in the orientation shown in the illustration below. A 100mm potentiometer is shown for reference, but the board will accept either a 60mm or 100mm potentiometer.

Make sure the potentiometer firmly inserted and flush before soldering all connections. It is important to also solder the two chassis connections, as these are necessary for proper shielding and function of the touch interface and add additional mechanical support.

You will then need to solder a wire or pin header between each motor lead and its accompanying motor pad, located on the part of the board that extends downwards by the motor.

The black arrows in the illustration below shows all solder connections that need to be made (11 in total).

If you are using a potentiometer with 60mm travel, you should now carefully cut or break off the part of the circuit board that extends beyond the length of this potentiometer. The board has a scoring line marked with the text *PSM60 BREAK LINE*, showing where the board should be detached.



HEADERS

Once you've installed the board, you will likely want to solder a connector to the board.

For most applications it is only necessary to install the middle 10-pin connector, as all other pins are reserved or for special applications only. All connector pads have a standard 2.54mm(0.1”) pitch and you can use any compatible header that fits your application.

We typically recommend an angled male pin-header unless your application has special requirements.

If you wish to use the QWIIC system to connect the board to a microcontroller or daisy chain multiple devices, now is a good time to install the QWIIC JST connectors. Please make sure to properly solder the chassis leads on each side of the connector, to ensure that the connector does not break off and damage the board.

SERIAL INTERFACE

OVERVIEW

The DSMP1602 operates only as a slave device on the I²C bus. Connections to the bus are made via the open-drain I/O lines ***SDA*** and ***SCL***. External pull-up resistors must be employed on the bus. DSMP1601 support the transmission protocol for *fast* (up to 400kHz) modes.

To access the DSMP1602, the master must first address slave devices via a slave address byte. The slave address byte consists of seven address bits and a direction bit indicating the intent of executing a read or write operation.

The DSMP1602 features four address pins (***A0 – A4***) for address selection, see the **ADDRESS SELECTION** section for more information on available addresses and their configuration.

READING/WRITING

Accessing a particular register on the DSMP1602 is accomplished by writing the appropriate value to the pointer register. The value for the pointer register is the first byte transferred after initiating a write operation. Every write operation to the DSMP1602 requires a value for the pointer register to be transferred.

During a write operation, all following bytes will be written to the register determined by the pointer register.

When reading from the DSMP1602, the last value stored in the pointer register from a write operation will be used to determine which register is read. To change the pointer register for a read operation, a new value must be written to the pointer register. This is accomplished by issuing a write operation, followed by the pointer register byte and no additional data. The master can then issue a read operation to read the specified register.

POINTER REGISTER

Typically, the pointer register will automatically increment after each successful register read or write. This allows reading or writing of consecutive registers without the need to issue a write operation and pointer register byte for each register.

Some registers however, termed streaming registers, will not increment the register pointer after a successful read or write.

Since each write operation requires a pointer register byte to be sent first, multiple registers can only be written in a single I²C transaction.

For read operations the register pointer will be maintained between transactions and it is possible to read multiple consecutive registers over multiple transactions.

When automatically incrementing the pointer register into a range of non-existing register addresses, the pointer register will automatically skip to the next valid register address.

However, if the pointer register is written with a non-existing register address, it will be reset to 0.

16-BIT REGISTERS

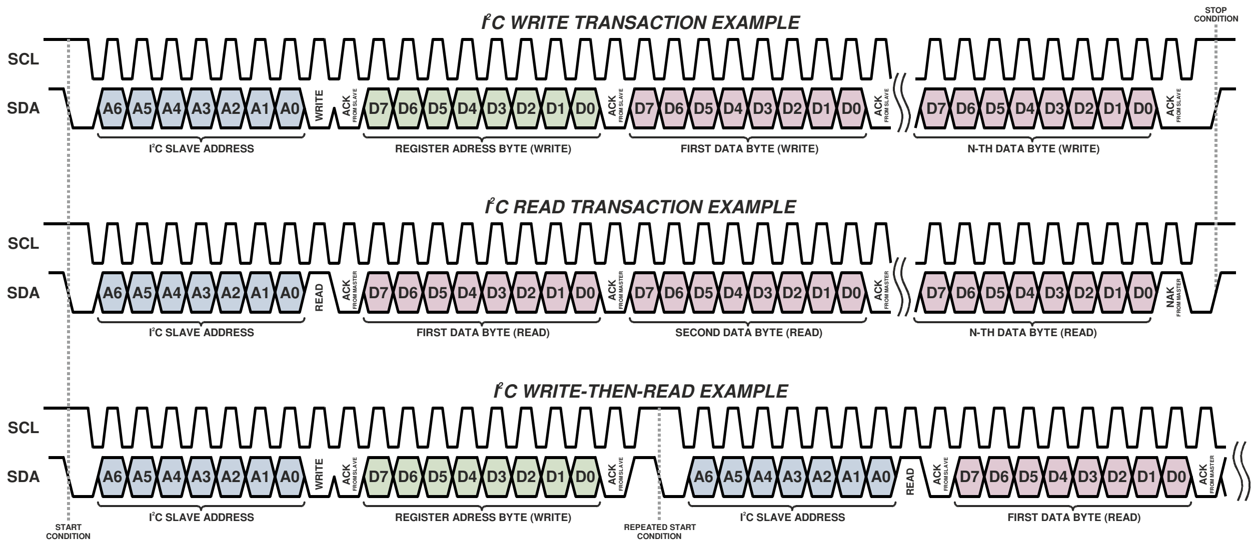
The DSMP1602 contains several 16-bit register that requires special care when reading and writing.

16-bit data is always transferred ***BIG ENDIAN*** (most significant byte first), also known as network order.

When reading a 16-bit register, always read the most significant byte (MSB) first. The DSMP1602 will latch the least significant byte of the register into a temporary storage when this occurs. When reading the least significant byte (LSB) no actual register access is performed, instead the temporary storage is returned. This prevents incorrect data transfers if the register contents change in between the time of reading the MSB and reading the LSB.

Similarly, when writing a 16-bit register, always write the MSB first. When this occurs no actual register access takes place, instead the MSB will be latched into a temporary storage. When the LSB is written, the MSB from the temporary storage and LSB will be used to write the full 16-bit register.

When reading or writing 16-bit registers always access the MSB and LSB together, in direct succession. These two reads or writes may be split over multiple transactions, but no other registers should be accessed between the two. Doing so invokes undefined behavior.

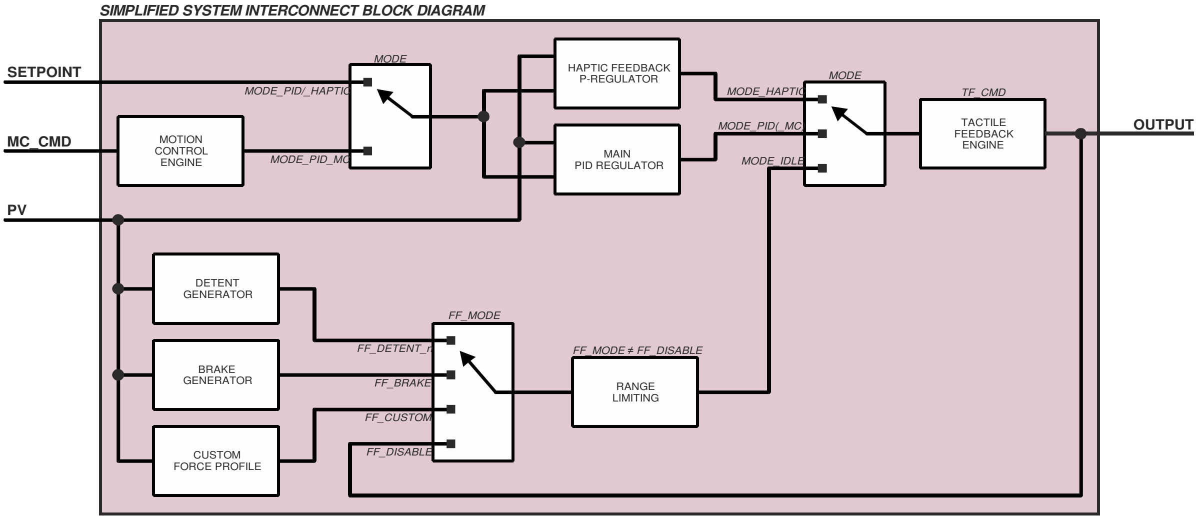


DS PROTOTYP

FUNCTIONAL DESCRIPTION

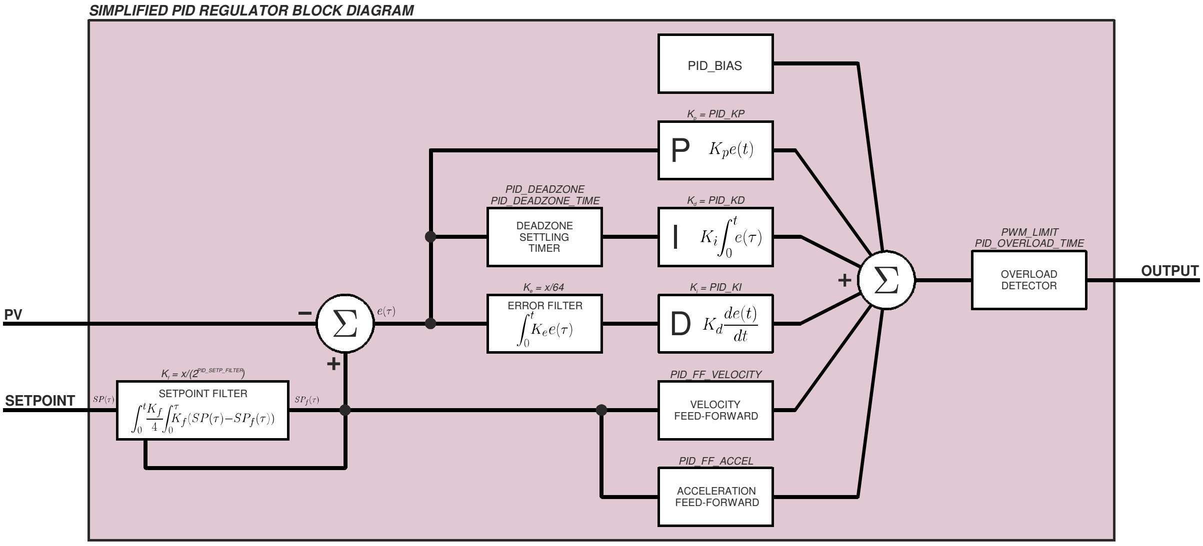
SYSTEM INTERCONNECT

The DSPSM1602 consists of a number of separate and interconnected modules that operate in conjunction with each other to perform all of the tasks related to movement and various feedback modes. The system interconnect fabric is illustrated by the following diagram.



MAIN PID REGULATOR

The DSPSM1602 main regulator is a typical PID regulator with the addition of bias and feed-forward terms, as well as filters specific to accurately regulating and controlling a system based on analog feedback from a potentiometer track. The functionality of the regulator is illustrated by the following diagram.



PROGRAMMING FORCE FEEDBACK MODES

DETENTS

Programming detents consists of two parts – programming detent shape, and programming detent placement.

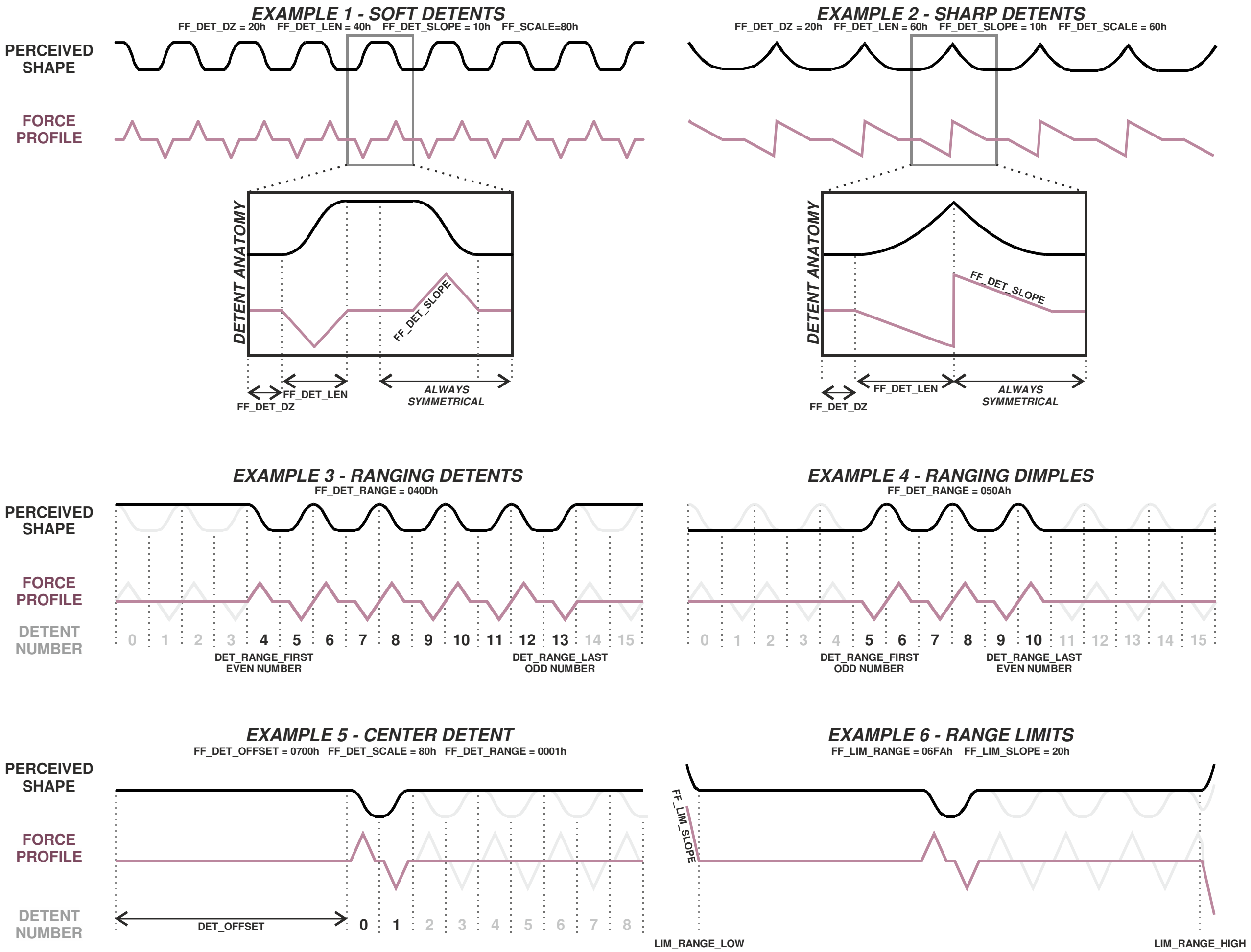
Detent shape is defined by a deadzone (**FF_DET_DZ**) – the size of the “valley” at the bottom of each detent, the length of the inclines surrounding each detent (**FF_DET_LEN**) and the slope of those inclines (**FF_DET_SLOPE**). In addition, the force feedback mode (**FF_MODE**) defines whether the valleys roll off softly (**FF_DETENT_SOFT**) or sharply (**FF_DETENT_SHARP**). If the values of **FF_DET_DZ** and **FF_DET_LEN** add upp to less than 80h a flat region will be present between the detents, as illustrated in *Example 1*. In *Example 2*, this is not the case – and the peak has no flat region.
Detent placement is specified by a scale (**FF_DET_SCALE**) and an offset (**FF_DET_OFFSET**). In *Example 1* scale is defined for 8 detents, while *Example 2* shows scale defined for 6 detents. *Example 5* shows the use of offset to move detents in either direction. In addition, it is possible to limit which detents are active (**FF_DET_RANGE**), which is illustrated by *Example 3-5*.

LIMITED MOVEMENT RANGE

It is also possible to program a limited movement range (**FF_LIM_RANGE**) as shown by *Example 6*. This is applied separately and can be combined with all force feedback modes. It is also possible to define the force applied (**FF_LIM_SLOPE**) to counter movement beyond the programmed limits.

CUSTOM FORCE PROFILE

All examples show their corresponding force profile. This profile shows what data needs to be uploaded (**PROFILE_UPLOAD**) to the custom force profile to replicate the shown behavior. Each data point in the force profile specifies how much force should be applied when the slider is in that specific location.



REGISTER LAYOUT

REGISTER	ADDRESS	FORMAT	RANGE	NOTES
VERSION	00h	8-bit		Read only, Streaming register[1]
PWM_LIMIT	10h	8-bit (unsigned)	0h-FFh	
PV_FILTER	11h	8-bit (unsigned)	0h-1Fh	
HAPTIC_KP	12h	8-bit (unsigned)	0h-FFh	
PID_KP	20h	8-bit (unsigned)	0h-FFh	
PID_KI	21h	8-bit (unsigned)	0h-FFh	
PID_KD	22h	8-bit (unsigned)	0h-FFh	
PID_BIAS	23h	8-bit (unsigned)	0h-FFh	
PID_DEADZONE	24h	16-bit (unsigned)	0h-1FFh	
PID_DEADZONE_TIME	26h	16-bit (unsigned)	0h-FFFFh	
PID_OVERLOAD_TIME	28h	16-bit (unsigned)	0h-FFFFh	
PID_FF_VELOCITY	30h	8-bit (unsigned)	0h-FFh	
PID_FF_ACCEL	31h	8-bit (unsigned)	0h-FFh	
PID_SETP_FILTER	32h	8-bit (unsigned)	0h-Fh	
MC_VELOCITY	40h	16-bit (unsigned)	0h-FFFFh	
MC_ACCEL	42h	16-bit (unsigned)	0h-FFFFh	
MC_SETTLE_MIN	44h	16-bit (unsigned)	0h-FFFh	
MC_SETTLE_MAX	46h	16-bit (unsigned)	0h-FFFh	
FF_MODE	50h	8-bit		
FF_DET_DZ	51h	8-bit (unsigned)	0h-FFh	
FF_DET_SLOPE	52h	8-bit (unsigned)	0h-FFh	
FF_DET_LEN	53h	8-bit (unsigned)	0h-FFh	
FF_DET_OFFSET	54h	16-bit (signed)	0h-FFFFh	
FF_DET_SCALE	56h	16-bit (unsigned)	0h-3FFFh	
FF_DET_RANGE	58h	16-bit		
FF_LIM_RANGE	5Ah	16-bit		
FF_LIM_SLOPE	5Bh	8-bit (unsigned)	0h-FFh	
FF_BRK_DRAG	5Dh	8-bit (unsigned)	0h-FFh	
FF_BRK_KP	5Eh	8-bit (unsigned)	0h-FFh	
SYSTEM	80h	8-bit		
STATE	81h	8-bit		
OUTPUT	82h	16-bit (signed)	FF00h-100h	Streaming register[1]
SETPOINT	84h	16-bit (unsigned)	0h-FFFh	Streaming register[1]
PV_FILTERED	8Ah	16-bit (unsigned)	0h-FFFh	Read only, streaming register[1]
PV	8Ch	16-bit (unsigned)	0h-FFFh	Read only, streaming register[1]
PID_ERROR	8Eh	16-bit (signed)	F001h-FFFh	Read only, streaming register[1]
MC_CMD	A0h	16-bit		Command, streaming register[1]
TF_CMD	A8h	8-bit		Command, streaming register[1]
PROFILE_SETUP	B1h	8-bit		Command register
PROFILE_UPLOAD	B2h	8-bit		Streaming register[1]
SCRATCH0	FCh	8-bit		
SCRATCH1	FDh	8-bit		
SCRATCH2	FEh	8-bit		
SCRATCH3	FFh	8-bit		

[Note1] Streaming registers will not auto-increment the register pointer.

VERSION

Address: 00h

Bit	7	6	5	4	3	2	1	0
Read	**VERSION_CHAR**							
Write	**VERSION_RST**							
Reset	x	x	x	x	x	x	x	x

Field	Description
7-0 VERSION_CHAR VERSION_RST	Version String Contains the string ”DSMP001” followed by a 00h. The register has an internal indexing counterand each time the register is read the next successive character from the string is returned. Write any data to the register to reset the internal indexing counter.

PWM_LIMIT

Address: 10h

Bit	7	6	5	4	3	2	1	0
Read/Write	**PWM_LIMIT**							
Reset	1	1	0	1	0	1	0	0

Field	Description
7-0 PWM_LIMIT	Motor PWM drive limit Controls the maximum amount of power to the motor. For example, if the drive voltage is 12V setting this value to 127 (50%) vill limit the drive voltage to 6V.

PV_FILTER

Address: 11h

Bit	7	6	5	4	3	2	1	0
Read/Write				**PV_FILTER**				
Reset	x	x	x	0	0	1	1	0

Field	Description
4-0 PV_FILTER	The amount of filtering applied to the filtered PV register, **PV_FILTERED.** The filter is a single-pole low-pass filter with a sampling rate of 100kHz and a decay of $1-(1/(2^{PV\_FILTER}))$

HAPTIC_KP

Address: 12h

Bit	7	6	5	4	3	2	1	0
Read/Write	**HAPTIC_KP**							
Reset	0	0	0	0	1	0	1	0

Field	Description
7-0 HAPTIC_KP	The P-term of the simplified haptic feedback P regulator, used when **MODE** = **MODE_HF.**

PID_KP

Address: 20h

Bit	7	6	5	4	3	2	1	0
Read/Write	PID_KP							
Reset	0	0	0	1	1	1	1	1

Field	Description
7-0 PID_KP	The P-term of the main PID regulator.

PID_KI

Address: 21h

Bit	7	6	5	4	3	2	1	0
Read/Write	PID_KI							
Reset	0	0	0	0	0	1	0	0

Field	Description
7-0 PID_KI	The I-term of the main PID regulator.

PID_KD

Address: 22h

Bit	7	6	5	4	3	2	1	0
Read/Write	PID_KD							
Reset	0	0	0	0	1	0	0	0

Field	Description
7-0 PID_KD	The D-term of the main PID regulator.

PID_BIAS

Address: 23h

Bit	7	6	5	4	3	2	1	0
Read/Write	PID_BIAS							
Reset	0	0	0	0	0	0	0	0

Field	Description
7-0 PID_BIAS	The bias of the main PID regulator. Bias is a signed value between -127 and +127 representing -50% to +50% of the available motor voltage. Bias is added at the very end of the PID calculation to account to asymmetrical loading.

PID_DEADZONE

Address: 24h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write								PID_DEADZONE								
Reset	x	x	x	x	x	x	x	0	0	0	1	0	1	0	0	0

Field	Description
8-0 PID_DEADZONE	The main PID regulator dead-zone. Used in conjunction with **PID_DEADZONE_TIME**. When the error stays less than this value for **PID_DEADZONE_TIME** the regulator will stop regulating to reduce noise, power and wear.

PID_DEADZONE_TIME

Address: 26h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write	PID_DEADZONE_TIME															
Reset	0	0	1	0	0	1	1	1	0	0	0	1	0	0	0	0

Field	Description
15-0 PID_DEADZONE_TIME	The dead-zone time-out in tens of microseconds (1 = 10µS). See **PID_DEADZONE** for more information.

PID_OVERLOAD_TIME

Address: 28h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write	PID_OVERLOAD_TIME															
Reset	0	0	1	0	0	1	1	1	0	0	0	1	0	0	0	0

Field	Description
15-0 PID_OVERLOAD_TIME	The overload timer of the main PID regulator. If the regulator stays at maximum output as defined by **PWM_LIMIT** for more than this time, the **OVERLOAD** flag will be set in the **STATE** register. The motion controller may monitor this bit and abort a movement depending on its current configuration. The unit is tens of microseconds (1 = 10µS).

PID_FF_VELOCITY

Address: 30h

Bit	7	6	5	4	3	2	1	0
Read / Write	PID_FF_VELOCITY							
Reset	0	0	1	1	1	1	1	0

Field	Description
7-0 PID_FF_VELOCITY	The velocity (speed) feed-forward term of the main PID regulator. Velocity is approximated from changes to the filtered setpoint value.

PID_FF_ACCEL

Address: 31h

Bit	7	6	5	4	3	2	1	0
Read / Write	PID_FF_ACCEL							
Reset	0	0	0	0	0	1	0	0

Field	Description
7-0 PID_FF_ACCEL	The acceleration feed-forward term of the main PID regulator. Acceleration is approximated from changes to the filtered setpoint value.

PID_SETP_FILTER

Address: 32h

Bit	7	6	5	4	3	2	1	0
Read / Write					PID_SETP_FILTER			
Reset	x	x	x	x	0	1	1	0

Field	Description
3-0 PID_SETP_FILTER	Filtering of the setpoint for the internal regulators. The filter is a two-pole low-pass filter used to smooth out step responses from the internal motion controller, and when external setpoint control is in use. The filter needs to be increased for applications where the setpoint register is updated less frequently. Suitable update rate of the setpoint register is roughly $1000\text{Hz}^{\text{PID\_SETP\_FILTER}}$.

MC_VELOCITY

Address: 40h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write	MC_VELOCITY															
Reset	0	1	0	0	1	1	1	0	0	0	1	0	0	0	0	0

Field	Description
15-0 MC_VELOCITY	Maximum velocity (speed) of the internal motion controller, used for positioning commands.

MC_ACCEL

Address: 42h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write	MC_ACCEL															
Reset	0	0	0	0	0	0	0	1	1	1	1	1	0	1	0	0

Field	Description
15-0 MC_ACCEL	Maximum acceleration of the internal motion controller, used for positioning commands.

MC_SETTLE_MIN

Address: 44h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write					**MC_SETTLE_MIN**											
Reset	x	x	x	x	0	0	0	0	0	1	1	0	0	1	0	0

Field	Description
11-0 MC_SETTLE_MIN	Minimum settling time of the internal motion controller. The motion controller will always allow the main PID regulator this amount of time to settle at the end of each move. If the PID regulator then considers itself stable (see **PID_DEADZONE** and **PID_DEADZONE_TIME**) the move is finished and the motion controller shuts down. The unit is milliseconds.

MC_SETTLE_MAX

Address: 46h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write					**MC_SETTLE_MAX**											
Reset	x	x	x	x	0	0	0	0	1	1	1	1	1	0	1	0

Field	Description
11-0 MC_SETTLE_MAX	Maximum settling time of the internal motion controller. If the PID regulator does not consider itself stable (see **PID_DEADZONE** and **PID_DEADZONE_TIME**) after this amount of time, the move is cancelled and the **UNSETTLED** flag is set in the **STATE** register. The unit is milliseconds.

FF_MODE

Address: 50h

Bit	7	6	5	4	3	2	1	0
Read / Write						**FF_MODE**		
Reset	x	x	x	x	x	0	0	0

Field	Description
3-0 FF_MODE	Force feedback mode. 0b000 Disabled (**FF_DISABLE**) 0b001 Enabled - limits are active (**FF_ENABLED**) 0b010 Brake mode (**FF_BRAKE**) 0b011 Custom profile (**FF_CUSTOM**) 0b100 Detents, soft (**FF_DETENT_SOFT**) 0b101 Detents, sharp (**FF_DETENT_SHARP**) 0b110 reserved 0b111 Detents, custom profile (**FF_DETENT_CUSTOM**)

FF_DET_DZ

Address: 51h

Bit	7	6	5	4	3	2	1	0
Read Write	**FF_DET_DZ**							
Reset	0	0	1	1	0	0	0	0

Field	Description
7-0 FF_DET_DZ	Size of the dead-zone / "bottom valley" of each detent. Used to define detent shape (when **FF_MODE** is any of **FF_DETENT_n**).

FF_DET_SLOPE

Address: 52h

Bit	7	6	5	4	3	2	1	0
Read Write	**FF_DET_SLOPE**							
Reset	0	0	0	1	0	0	0	0

Field	Description
7-0 FF_DET_SLOPE	Slope of the inclines surrounding each detent. Used to define detent shape (when **FF_MODE** is any of **FF_DETENT_n**).

FF_DET_LEN

Address: 53h

Bit	7	6	5	4	3	2	1	0
Read Write	**FF_DET_LEN**							
Reset	0	1	1	1	0	0	0	0

Field	Description
7-0 FF_DET_LEN	Sets the length of the inclines surrounding each detent. Used to define detent shape (when **FF_MODE** is any of **FF_DETENT_n**).

FF_DET_OFFSET

Address: 54h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read Write	**FF_DET_OFFSET**															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15-0 FF_DET_OFFSET	Offset (move) detents (when **FF_MODE** is any of **FF_DETENT_n**). As an example, a value of 100h represents moving detents by 1/16th of the full scale. This is a signed register and may be negative.

FF_DET_SCALE

Address: 56h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write			**FF_DET_SCALE**													
Reset	x	x	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Field	Description
13-0 FF_DET_SCALE	Scale detents (when **FF_MODE** is any of **FF_DETENT_n**). As an example, a value of 100h represents 16 detents while a value of 400h represents 4 detents.

FF_DET_RANGE

Address: 58h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write	**DET_RANGE_FIRST**								**DET_RANGE_LAST**							
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Field	Description
15-8 DET_RANGE_FIRST	Detent range – first detent (when **FF_MODE** is any of **FF_DETENT_n**). First detent to enable, detents with a lower number than this will not be active. Each detent is split into two parts, each with their own detent number. The first detents left slope is numbered 0, its right slope numbered 1. The second detent is 2 and 3, and so on.
7-0 DET_RANGE_LAST	Detent range – last detent Last detent to enable, detents with a lower higher than this will not be active. See **DET_RANGE_FIRST** for information on detent numbering.

FF_LIM_RANGE

Address: 5Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write	**LIM_RANGE_LOW**								**LIM_RANGE_HIGH**							
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Field	Description
15-8 LIM_RANGE_LOW	Limited movement range – low limit (when **FF_MODE** is not **FF_DISABLE**). If the user moves the actuator below this limit a force defined by **FF_LIM_SLOPE** will be applied to counter further movement in this direction. Values 0-255 represents the full range of the actuator.
7-0 LIM_RANGE_HIGH	Limited movement range – high limit (when **FF_MODE** is not **FF_DISABLE**). If the user moves the actuator above this limit a force defined by **FF_LIM_SLOPE** will be applied to counter further movement in this direction. A value of 0-255 represents the full range of the actuator.

FF_LIM_SLOPE

Address: 5Bh

Bit	7	6	5	4	3	2	1	0
Read / Write	**FF_LIM_SLOPE**							
Reset	0	0	0	0	0	0	0	0

Field	Description
7-0 FF_LIM_SLOPE	Limited movement range – slope (when **FF_MODE** is not **FF_DISABLE**). Sets the amount of force applied to counter movements beyond the limits specified in **FF_LIM_RANGE**.

FF_BRK_DRAG

Address: 5Dh

Bit	7	6	5	4	3	2	1	0
Read / Write	**FF_BRK_DRAG**							
Reset	0	0	1	0	0	0	0	0

Field	Description
7-0 FF_BRK_DRAG	Force brake - drag (when **FF_MODE** is **FF_BRAKE**). Sets the amount of drag applied when trying to move the actuator.

FF_BRK_KP

Address: 5Eh

Bit	7	6	5	4	3	2	1	0
Read / Write	**FF_BRK_KP**							
Reset	0	0	1	0	0	0	0	0

Field	Description
7-0 FF_BRK_KP	Force brake – P term (when **FF_MODE** is **FF_BRAKE**). Sets the amount of force applied to counter movements the actuator.

SYSTEM

Address: 80h

Bit	7	6	5	4	3	2	1	0
Read	**RST**		**FACT**	**LOAD**	**SAVE**	**CALT**		**GCAE**
Write								**GCAE**
Reset	x	x	x	x	x	x	x	0

Field	Description
7 RST	Write a one to this bit to restart the device.
5 FACT	Load factory default configuration. After loading factory default configuration **STATE** will be reset and **FF_MODE** will be set to **FF_NONE**.
4 LOAD	Load configuration from non-volatile memory. If a valid configuration does not exist in non-volatile memory, the factory defaults will be loaded instead. After loading the configuration **STATE** will be reset and **FF_MODE** will be set to **FF_NONE**.
3 SAVE	Store current configuration in non-volatile memory. Write a 1 to this bit to store the current configuration in non-volatile memory. Once a configuration has been written to non-volatile memory it will be loaded automatically on start-up. To revert a stored configuration back to default, issue a write to **SYSTEM** with both **SAVE** and **FACT** set. The non-volatile memory has a write endurance of 10.000 cycles. To prevent wear in a fault condition the following rules apply: 1) The configuration can not be saved until one second after the device has become active, either from power-up or a reset. 2) After successfully storing a configuration, another **SAVE** command can not be issued for 5 seconds.
2 CALT	Calibrate Touch Sensor In the event that the touch sensor has lost environmental tracking and starts reporting an incorrect touch state, write a 1 to this bit to issue a recalibration of the touch controller.
0 GCAE	I^2C General Call Address Enable When this bit is set, the device respond to the 00h general call address.

STATE

Address: 81h

Bit	7	6	5	4	3	2	1	0
Read	**INI**	**OVL**	**ABO**	**UNS**	**TACT**	**TOUCH**	**MODE**	
Write	**CINI**	**COVL**	**CABO**	**CUNS**			**MODE**	
Reset	1	0	0	0	0	x	0	0

Field	Description
7 INI/CINI	(Re-)initialized. This flag will always be set when the system is powered up, or after a reset, brown-out or other reason. Write a 1 to this location to clear the flag.
6 OVL/COVL	PID regulator overload. The PID regulatorhas detected an overload condition. See **PID_OVERLOAD_TIME** for more information. Write a 1 to this location to clear the flag.
5 ABO/CABO	Motion Controller Aborted. The motion controller has aborted a move due to overload or touch. Check **OVL** to determine the reason and see **MC_CMD** for more information on movements and error reporting. Write a 1 to this location to clear the flag.
4 UNS/CUNS	Motion Controller Unsettled. The motion controller finished while the PID regulator was unsettled. See **PID_DEADZONE, MC_SETTLE** and **MC_CMD** for more information. Write a 1 to this location to clear the flag.
3 TACT	Tactile event is in progress.
2 TOUCH	Actuator is being touched.
1-0 MODE	Current working mode. When writing the register, **MODE** will be ignored the written byte has any flags set – to prevent switching to idle when clearing flags. 0b00 Idle & force feedback modes (**MODE_IDLE**) 0b01 Haptic feedback P-regulation (**MODE_HAPTIC**) 0b10 PID regulation (**MODE_PID**) 0b11 PID+Motion Controller (**MODE_PID_MC**) Mode **MODE_PID_MC** is set automatically when issuing move commands and should never be set using this register. However, it is possible to abort the currently executing move by setting this register to something other than **MODE_PID_MC** while the move is in progress. When switching mode to **MODE_PID** from another mode the **UNS** and **OVL** flags will be cleared and **SETPOINT** will be loaded from **PV. SETPOINT** will also be loaded from **PV** when switching to **MODE_HAPTIC** from another mode.

OUTPUT

Address: 82h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write	**OUTPUT**															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
15-0 OUTPUT	Motor output register (torque). When **MODE** is **MODE_IDLE** and **FFMODE** is **FF_DISABLED** this register can be used to control motor torque directly. In all other configurations this register is read-only and contains the currently applied motor torque. The range is -256 to +256.

SETPOINT

Address: 84h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read / Write					**SETPOINT**											
Reset	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0

Field	Description
11-0 SETPOINT	Setpoint register for PID position control. When **MODE** is **MODE_PID** or **MODE_HF** this register can be used to directly control the setpoint (wanted position) of the internal regulators. When **MODE** is **MODE_PID_MC** this register is controlled by the motion controller and while not strictly read-only it is not recommended to write this register while the motion controller is active as this may cause glitches in movement.

PV_FILTERED

Address: 8Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read					**PV_FILTERED**											
Write																
Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Field	Description
11-0 PV_FILTERED	Potentiometer Value/Process variable – filtered. This register contains the current position of the slider, filtered according to **PV_FILTER**.

PV

Address: 8Ch

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read					**PV**											
Write																
Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Field	Description
11-0 PV	Potentiometer Value/Process variable. This register contains the current position of the slider, unfiltered.

PID_ERROR

Address: 8Eh

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	**PID_ERROR**															
Write																
Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Field	Description
15-0 PID_ERROR	Current error term for the internal regulators. This register contains the error term as seen by the internal PID regulators in the range -4095 to +4095.

MC_CMD

Address: A0h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	**MCA**	**OVL**	**ABO**	**UNS**	**PV_FILTERED**											
Write	**ATO**	**AOV**			**MOVE_CMD**											
Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Field	Description
15 ATO	Abort on touch. The issued move will be aborted if the actuator is touched.
14 AOV	Abort on overload. The issued move will be aborted if the PID regulator registers an overload condition. See **PID_OVERLOAD_TIME** for more information.
11-0 MOVE_CMD	Movement command. Writing to this register initiates a movement command using the motion controller to the position specified by **MOVE_CMD**. **MODE** will be switched to **MODE_PID_MC** for the duration of the movement and switched back to its original state once the move has completed. The move will adhere to constraints set by **MC_VELOCITY** and **MC_ACCELERATION**. It is possible to issue another move command while the first is still being executed. The first move will abort and the new move will be initiated immediately.
15 MCA	Motion Controller Active. This flag will be set for the duration of the motion controller activation and will clear automatically when the movement is completed. This flag is defined by **MODE** = **MODE_PID_MC**.
14 OVL	See **STATE** - **OVL**.
13 ABO	See **STATE** - **ABO**.
12 UNS	See **STATE** - **UNS**.
11-10 PV_FILTERED	See **PV_FILTERED**.

TF_CMD

Address: A8h

Bit	7	6	5	4	3	2	1	0
Read	**TF_TIMER**							
Write	**TF_STRENGTH**			**TF_SPEED**		**TFNTF**	**TF_DURATION**	
Reset	x	x	x	x	x	x	x	x

Field	Description
7-5 TF_STRENGTH	Used to trigger a tactile feedback event. Sets the strength of the event, a value of 0 will cancel any currently active event.
4-3 TF_SPEED	Sets the vibration speed of the event.
2 TFNTF	Force the event, even if the controller is not being touched.
1-0 TF_DURATION	Sets the duration of the event.
7-0 TF_TIMER	If a tactile feedback event is in progress this register contains the internal counter, counting down until the end of the event. Will return 0 when no tactile feedback event is in progress.

PROFILE_SETUP

Address: B1h

Bit	7	6	5	4	3	2	1	0
Read / Write						**PROFILE_BITS**		
Reset	0	0	0	0	0	0	0	0

Field	Description
2-0 PROFILE_BITS	Custom force profile set up register. When this register is written it will clear the currently uploaded force profile and prepare for a new upload. After writing this field, write $2^{(PROFILE\_BITS+1)}$ bytes to **PROFILE_UPLOAD** to upload the new force profile. Once this has been done it will be possible to use force feedback modes **FF_CUSTOM** and **FF_DETENT_CUSTOM** by setting **FF_MODE** appropriately.

PROFILE_UPLOAD

Address: B2h

Bit	7	6	5	4	3	2	1	0
Read	**RECEIVED_BYTES**							
Write	**PROFILE**							
Reset	x	x	x	x	x	x	x	x

Field	Description
7-0 PROFILE	Upload register for custom force profile. After writing **PROFILE_BITS** to configure a new force profile, this register needs to be written $2^{(PROFILE\_BITS+1)}$ times to complete the upload process.
RECEIVED_BYTES	Number of bytes received since the last write to **PROFILE_SETUP**. Can be used to check if the profile was successfully uploaded. Note that after uploading a full 256-byte profile, this register will show 0 due to integer roll-over.

SCRATCHn,

Address: FCh-FFh

Bit	7	6	5	4	3	2	1	0
Read / Write	**SCRATCHn**							
Reset	1	1	1	1	1	1	1	1

Field	Description
7-0 SCRATCHn	Four bytes of user scratch RAM. Will be stored and recalled from non-volatile memory together with all other configurable parameters.