NXP Semiconductors
Application Note

Document Number: AN5187
Rev. 1.0, 7/2016

PT2000: Three injectors with freewheeling and DCDC

1 Introduction

This application note examines an example of a three cylinder internal combustion engine (ICE) injector drive. The high-voltage is provided by a boost converter managed by the PT2000. It describes a typical hardware topology and related software example to drive three injectors managed in three banks, allowing a full overlap and a DC/DC converter in resonant mode.

The PT2000 is supplied by a battery voltage between 9.0 V and 32 V. An external 5.0 V must be supplied to the VCC5 and the VCCIO pin, to internally supply the I/O buffers. In this example, the V_{CCP} voltage is internally generated to enable the drivers. If V_{BAT} voltage is set higher than 16 V then V_{CCP} has to be forced externally. The boost converter topology is defined to manage DC/DC. A Pi filter prevents circuitry disturbance propagation from the boost regulation area to battery line.

In this configuration, full overlap injection is possible, and to reduce power dissipation and show some capabilities of the PT2000 freewheeling low-sides are used instead of diode.

NXP analog ICs are manufactured using the SMARTMOS process, a combinational BiCMOS manufacturing flow integrating precision analog, power functions, and dense CMOS logic together on a single cost-effective die.

Contents

2 Application schematic

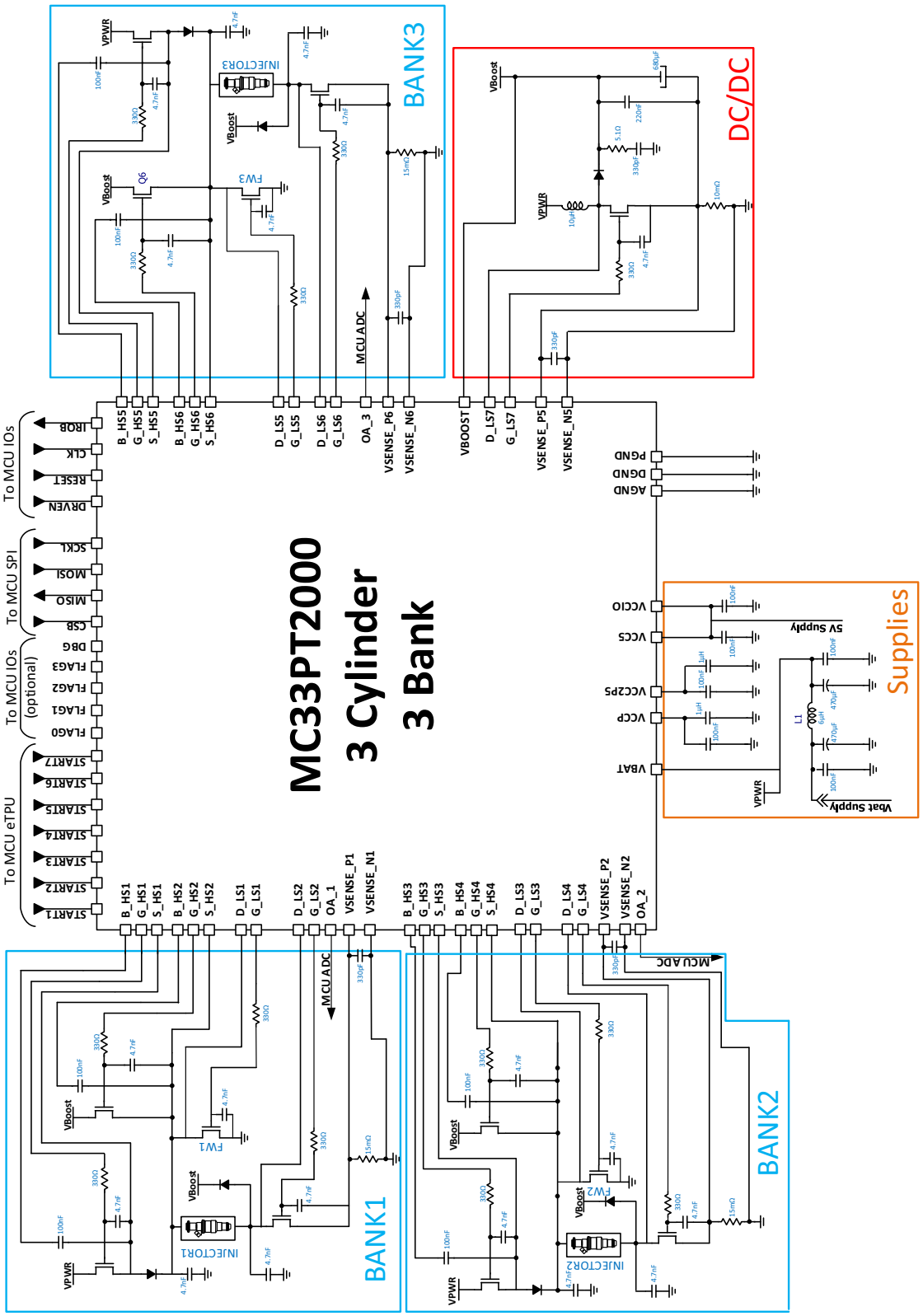

Figure 1. Typical four injector two bank application schematic

3 Application instructions

This topology can be used on the KITPT2000FRDM3C evaluation board. Register settings and microcode downloads can be achieved by using the KL25Z embedded on the KITPT2000FRDM3C. This topology manages three banks of three cylinders. Each bank is individually managed by one microcore, as follows:

- The bank # 1 is managed by the digital microcore Uc0Ch1
- The bank # 2 is managed by the digital microcore Uc1Ch1.
- The bank # 3is managed by the digital microcore Uc0Ch2.

The DCDC is managed by microcore Uc0Ch3. The following is the start-up sequence:

- Apply a battery voltage between 9.0 V and 16 V.
- Download the registers Channel Configuration, Main Configuration, IO Configuration, and then Diagnostic Configuration.
- Download the dedicated microcode in the Logic Channel 1, Logic Channel 2, and Logic Channel 3 Data RAM.
- Set to ‘1’ to the pre-flash enable bit and the en_dual_seq bit in the Flash_enable register of channel 1 (0x100), channel 2 (0x120), and channel 3 (0x140).

The registers configuration and the microcodes are detailed in the following chapters.

Once the DC/DC converter output has reached its nominal voltage, the injector drivers can be actuated with the STARTx pins. Each STARTx pin individually triggers each injector pin rising edge and stops actuation on the falling edge.

- START1 drives INJECTOR 1
- START2 drives INJECTOR 2
- START3 drives INJECTOR 3

Boost regulation is stopped during the injection boost phase.

4 Software requirements

4.1 Current profile management for injection

The current profile is managed to generate an initial high current through the injector. This high current slew rate minimizes the opening delay. This high current is maintained during a given time to ensure injector opening, then is decreased to maintain the injector open, up to the End Of Injection (EOI).

The code dedicated to the injection is loaded into the Code RAM 1. This code is executed independently by the microcores Uc0Ch1 and Uc1Ch1. Each one of the microcores generate a current profile, as described through the injector per the STARTx pin state as illustrated in Figure 2.

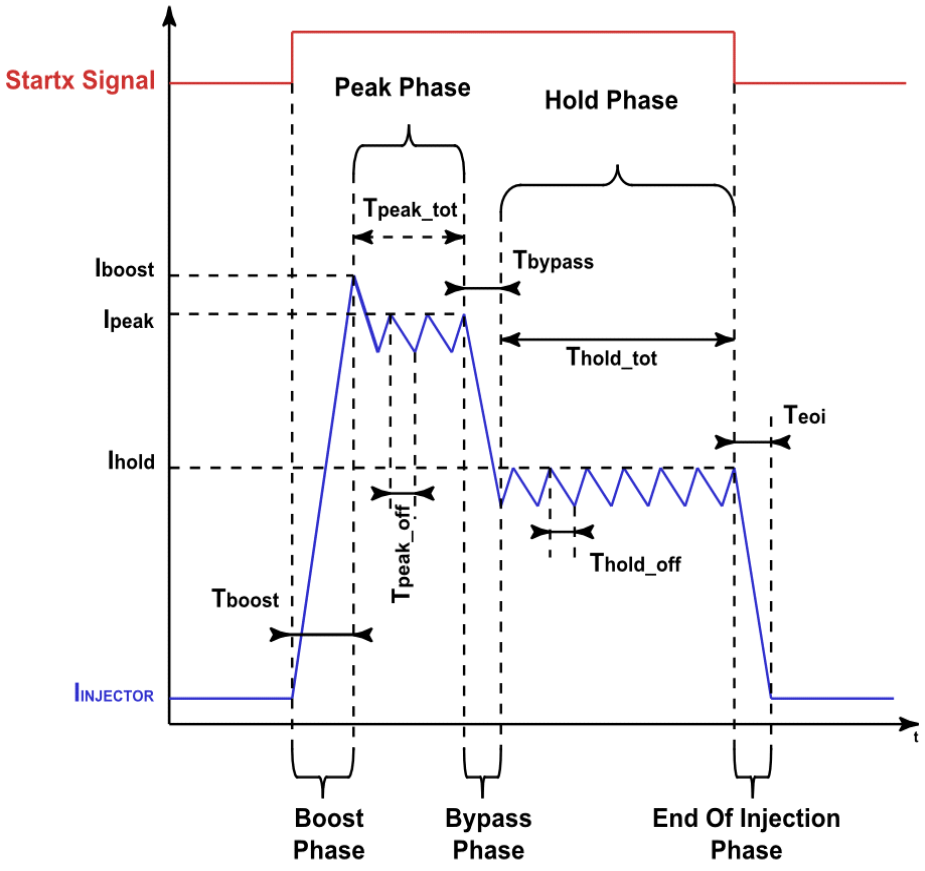

Figure 2. Typical peak and hold current profile

When a rising edge is detected on a STARTx pin, the injection starts with the injection Boost phase. This profile can be stopped at any time by detecting a falling edge on the STARTx pin. In this case, the End Of Injection phase is executed.

During the Boost phase, the corresponding low-side driver is simultaneously switched on with the high-side switch connected to the V_{BOOST} voltage. If the boost current target I_{BOOST} is reached, the high-side driver is switched off and the current recirculates for a fixed time (t_{PEAK_OFF}) through the diode connected to ground, then the Peak phase starts.

During the Peak phase, the high-side switch connected to V_{BAT} voltage is turned on. If the I_{PEAK} current target is met, the high-side driver is switched off and the current recirculates through the diode connected ground for the fixed time (t_{PEAK_OFF}). The high-side driver is then switched on again. This cycle repeats until the internal counter reaches its terminal value (t_{PEAK_TOT}), then the Bypass phase begins.

During the Bypass phase, all low-side and high-side switches are turned off. The current decays through the injector, the diode connected to ground, and the diode connected to V_{BOOST} for a fixed time (t_{BYPASS}). The Hold phase then starts.

During the Hold phase, the low-side driver is simultaneously switched on with the high-side switch connected to the V_{BAT} voltage. If the hold current target I_{HOLD} is reached, the high-side driver is switched off and the current recirculates through the diode connected to ground for a fixed time (t_{HOLD_OFF}). The high-side driver is switched on again, and the cycle repeats until the STARTx pin goes low or the internal counter reaches its terminal value (t_{HOLD_TOT} (time out)). The End Of Injection is forced if no falling edge is detected on the STARTx pin.

All the current thresholds and timings are accessed in the Data RAM. The typical values are in Table 1., but must be defined according to the injector used and the injection profile expected.

Table 1. Example of injection current profile key parameters (R_{SENSE} = 15 mΩ)

Parameter name	Description	Value
I_{BOOST}	Current threshold in Boost phase	14 A
I_{PEAK}	Current threshold in Peak phase (Depends on injectors type)	6.0 A
I_{HOLD}	Current threshold in Hold phase	3.0 A
t_{PEAK_OFF}	Fixed time for high-side switch off in Peak phase	10 µs
t_{PEAK_TOT}	Fixed time for end of Peak phase	500 µs
t_{BYPASS}	Fixed time for Bypass phase	20 µs
t_{HOLD_OFF}	Fixed time for high-side switch off in Hold phase	10 µs
t_{HOLD_TOT}	Fixed time for end of Hold phase (timeout)	10 ms

In the present case, most of the code branches (jump) are managed according to the counters end of count and the current threshold, by the means of the wait table. The wait table rows are affected, as shown in Table 2., and are changed according to the injection phase.

Table 2. Example of wait table definition

Phase	Boost phase	Peak phase	Bypass phase	Hold phase	EOI phase
Row 1	If startx goes low then jumps to EOI phase	If startx goes low, then jumps to EOI phase	If startx goes low, then jumps to EOI phase	If startx goes low, then jumps to EOI phase	-
Row 2	If the injection current reaches I_{BOOST} then jumps to Peak phase	If t_{PEAK_TOT} is reached, then jumps to Bypass phase	If t_{BYPASS} is reached, then jumps to Hold phase	If t_{HOLD_TOT} is reached, then jumps to EOI phase	-
Row3	-	if t_{PEAK_OFF} is reached, jumps to Peak On phase (sub phase)	-	if t_{HOLD_OFF} is reached, jumps to Hold On phase (sub phase)	-
Row 4	-	if I_{PEAK} is reached, jumps to Hold Off phase (sub phase)	-	if I_{HOLD} is reached, jumps to Hold Off phase (sub phase)	-
Row 5	-	-	-	-	-

A rising edge issues an injection start. A falling edge triggers the end of injection. In case of an overlap between two STARTx pins on the same bank, it is manage by the Smart Start function of the device. In this case, the first STARTx rising edge is considered. The second STARTx pin high-state is consider when the first actuation is finished. The action of the injection corresponding to the second STARTx pin is stopped when the second STARTx pin falling edge occurs.

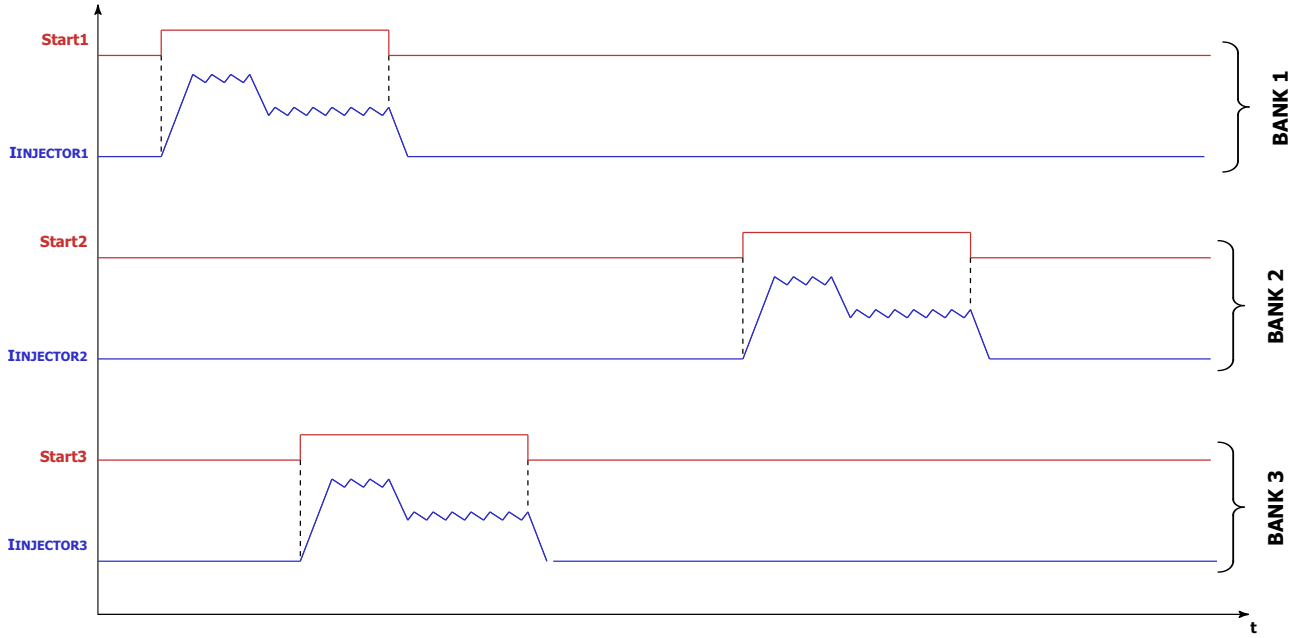

Figure 3. Actuation driven by STARTx pins with possible full overlap

4.1.1 SPI registers setup

The PT2000 registers are setup according their default states, unless defined by the following.

4.1.1.1 Main configuration registers

4.1.1.1.1 CLK configuration registers

A 1.0 MHz CLK has to be supplied from the main MCU to the PT2000. This CLK is then multiplied by a factor to generate the PLL, which is the clock for the microcore memories. In this example, the multiplicator factor chosen is 24, meaning a PLL at 24 MHz. This factor is set by PLL_Config register (1A7h).

Table 3. PLL_Config (1A7h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved														PLL_spread_disable	PLL_factor
Value	00000000000000														0	1

To run two microcores per channel, the **Clock_Prescaler** register (1A0h) must be setup with a ck_per value of 2 or 3. In this case, the main CLK (“ck”) is set to 6.0 MHz. ck = ck_sys/(ck_per +1) = 24 MHz/(3+1) = 6.0 MHz

Table 4. Clock_Prescaler register (1A0h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved										ck_per					
Value	-										000011					

4.1.1.1.2 Offset compensation CLK registers

To improve current accuracy offset compensation is enabled each time the microcore is in the idle state, which means the related start pin is low. To make this work, the CLK offset compensation must be set to a maximum of 500 kHz.

As shown in CLK configuration registers, page 6, the PLL ("ck_sys") is set to 24 MHz. To get a ck_ofscomp to 500 kHz. a prescaler set to 47d + 1d is needed.

Table 5. Ck_ofscmp_Prescaler(1A4h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved								ck_ofscmp_per							
Value	00000000								00101111							

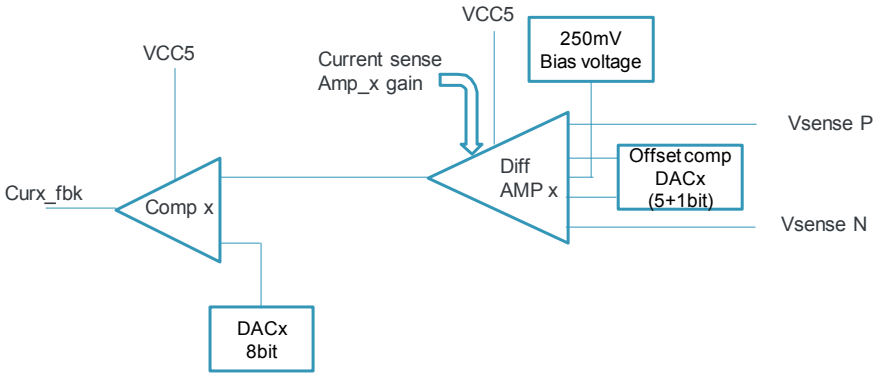

Figure 4. Offset compensation block diagram

As shown in Figure 4, a 5+1-bit DAC is use to compensate the offset. When the "stoc on" instruction runs the 8-bit DAC is automatically set to 253 mV and the 5+1 DAC for offset compensation tries to find the best value to get 250 mV. This is done during idle phase and shuts down as soon as the start pin goes high.

Each new offset compensation starts based on the result of the previous offset compensation run for this current measurement channel. If the offset compensation is stopped from the digital sequencer when the analog offset compensation is not finished, the procedure is aborted, maintaining the last compensation value reached when the procedure was interrupted. A full offset compensation takes 2.0 µs (500 kHz) x 31 steps = 62 µs.

4.1.2 Injection bank management register setup

The PT2000 registers are setup according their default states or setup automatically by the PT2000 IDE (i.e. checksum registers, code width), unless defined by the following.

4.1.2.1 IO configuration registers

These registers configure the crossbar switch to give access to output drivers, current sense to each microcore.

4.1.2.1.1 Output driver access

The Uc0Ch1 microcore must have access to the HS1, HS2, LS1, and LS2 pre-drivers.

Table 6. Out_acc_uc0_ch1 register (160h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Acc_uc0_ch1_ls8	Acc_uc0_ch1_ls7	Acc_uc0_ch1_ls6	Acc_uc0_ch1_ls5	Acc_uc0_ch1_ls4	Acc_uc0_ch1_ls3	Acc_uc0_ch1_ls2	Acc_uc0_ch1_ls1	Reserved	Acc_uc0_ch1_hs7	Acc_uc0_ch1_hs6	Acc_uc0_ch1_hs5	Acc_uc0_ch1_hs4	Acc_uc0_ch1_hs3	Acc_uc0_ch1_hs2	Acc_uc0_ch1_hs1
Value	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

In the same way, the Uc1Ch1 must have access to the pre-drivers HS3, HS4, LS3, and LS4.

Table 7. Out_acc_uc1_ch1 register (161h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Acc_uc1_ch1_ls8	Acc_uc1_ch1_ls7	Acc_uc1_ch1_ls6	Acc_uc1_ch1_ls5	Acc_uc1_ch1_ls4	Acc_uc1_ch1_ls3	Acc_uc1_ch1_ls2	Acc_uc1_ch1_ls1	Reserved	Acc_uc1_ch1_hs7	Acc_uc1_ch1_hs6	Acc_uc1_ch1_hs5	Acc_uc1_ch1_hs4	Acc_uc1_ch1_hs3	Acc_uc1_ch1_hs2	Acc_uc1_ch1_hs1
Value	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0

The Uc0Ch1 microcore has default access to the current sense block # 1, and to microcore Uc1Ch1 to the current sense block # 2. The corresponding registers content doesn't need to be changed.

In the same way, the Uc0Ch2 must have access to the pre-drivers HS5, HS6, LS5, and LS6.

Table 8. Out_acc_uc0_ch2 register (162h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Acc_uc1_ch1_ls8	Acc_uc1_ch1_ls7	Acc_uc1_ch1_ls6	Acc_uc1_ch1_ls5	Acc_uc1_ch1_ls4	Acc_uc1_ch1_ls3	Acc_uc1_ch1_ls2	Acc_uc1_ch1_ls1	Reserved	Acc_uc1_ch1_hs7	Acc_uc1_ch1_hs6	Acc_uc1_ch1_hs5	Acc_uc1_ch1_hs4	Acc_uc1_ch1_hs3	Acc_uc1_ch1_hs2	Acc_uc1_ch1_hs1
Value	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0

4.1.2.1.2 Current sense access

The following registers give access to current sense block to the specified microcore. The Uc0Ch1 microcore must have access to current sense 1 and Uc1Ch1 to current sense 2.

Table 9. Cur_block_access_part1 register (166h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	acc_uc1_ch1_curr6H_6Neg	acc_uc1_ch1_curr6L	acc_uc1_ch1_curr5H_5Neg	acc_uc1_ch1_curr5L	acc_uc1_ch1_curr4	acc_uc1_ch1_curr3	acc_uc1_ch1_curr2	acc_uc1_ch1_curr1	acc_uc0_ch1_curr6H_6Neg	acc_uc0_ch1_curr6L	acc_uc0_ch1_curr5H_5Neg	acc_uc0_ch1_curr5L	acc_uc0_ch1_curr4	acc_uc0_ch1_curr3	acc_uc0_ch1_curr2	acc_uc0_ch1_curr1
Value	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

The Uc0Ch2 microcore must have access to current sense 3.

Table 10. Cur_block_access_part2 register (167h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	acc_uc1_ch2_curr6H_6Neg	acc_uc1_ch2_curr6L	acc_uc1_ch2_curr5H_5Neg	acc_uc1_ch2_curr5L	acc_uc1_ch2_curr4	acc_uc1_ch2_curr3	acc_uc1_ch2_curr2	acc_uc1_ch2_curr1	acc_uc0_ch2_curr6H_6Neg	acc_uc0_ch2_curr6L	acc_uc0_ch2_curr5H_5Neg	acc_uc0_ch2_curr5L	acc_uc0_ch2_curr4	acc_uc0_ch2_curr3	acc_uc0_ch2_curr2	acc_uc0_ch2_curr1
Value	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

4.1.2.1.3 Freewheeling access

As mentioned previously, to reduce power dissipation and to show PT2000 capability, a freewheeling low-side is used instead of a diode. One of the features of the PT2000 is to control the FW low-side, automatically depending on the high-side command. Control of the freewheeling is done through microcore using the “stfw” instruction. There is no need to configure the Fw_external_request register (16Ah). High-side on V_{BAT} is the one used as a command, this high-side has to be ON even during the boost phase, to make sure the FW low-side is OFF. The following configuration is used in this example:

.

Table 11. Freewheeling link register

Freewheeling pre-driver output	Related pre-driver high-side
LS1	HS1
LS3	HS3
LS5	HS5

Table 12. Fw_link (169h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved	Flag3_fw_link	Flag2_fw_link	Flag1_fw_link	Flag0_fw_link	Reserved		Hs7_fw_link	Reserved	Ls7_fw_link	Ls6_fw_link	Ls5_fw_link	Ls4_fw_link	Ls3_fw_link	Ls2_fw_link	Ls1_fw_link
Reset	0	0	0	0	0	00		0	0	0	0	1	0	1	0	1

The time between the high-side and FW low-side (“dead time”) command is selectable by the SPI, to avoid cross conduction (refer to Table 20. High-side Output Configuration Register).

stfw instruction:

To automatically control the freewheeling low-side, the stfw auto instruction is used in the microcode. Shortcut 1 is considered as the high-side using freewheeling. In this case, HS1 uses LS1 as freewheeling the shortcut definition, for Uc0CH1 is: dfsct **hs1** hs2 ls2; in this case ls1 switches according to the hs1 command.

4.1.2.1.4 OAx settings (optional)

For safety purposes, it is possible to send an image of the current going through the load directly to the MCU through the OAx pins. The following settings map the current sense 1 to OA1 and current sense 2 to OA2.

Table 1. Oa_out1_config (197h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved									oa1_g1	oa_sel1			oa1_gain		oa1_en
Value	000000000									0	000			00		1

Table 13. Oa_out2_config (198h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved									oa2_g1	oa_sel2			oa2_gain		oa2_en
Value	000000000									0	000			00		1

If this feature is mandatory in the application, special care needs to be taken during the selection of the current for each bank. For example, if it needs to monitor on the OAx pins of each current sense at the same time, the following current sense needs to be selected:

-Bank 1 : current sense 1 mapped to OA1

-Bank 2: current sense 2 mapped to OA2

-Bank3 : current sense 5 mapped to OA3

4.1.2.2 Channel 1-2 configuration registers

The following registers are used to configure each channel used to control injectors.

4.1.2.2.1 Startx pin sensitivity registers

The injectors 1, 2, and 3 are driven according to the logic level of their respective STARTx pin. A high level on STARTx triggers the activation of the corresponding injector. A low level on the STARTx pin automatically stops the actuation, whatever the injection phase.

The Uc0Ch1microcore must be enabled by START1 pin, while the Uc1Ch1 microcore must be enabled by the START2 pin, and Uc0Ch2 must be enabled by START3 pin. Consequently, Start_config_reg_part1 register of the channel 1 (103h) must be setup, as shown in Table 14.

.

Table 14. Start_config_reg_part1 registers (103h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	start8_sens_uc1	start7_sens_uc1	start6_sens_uc1	start5_sens_uc1	start4_sens_uc1	start3_sens_uc1	start2_sens_uc1	start1_sens_uc1	start8_sens_uc0	start7_sens_uc0	start6_sens_uc0	start5_sens_uc0	start4_sens_uc0	start3_sens_uc0	start2_sens_uc0	start1_sens_uc0
Value	00		0	0	0	0	1	0	0	0	0	0	0	0	0	1

In the same way Start_config_reg_part1 register of the channel 2 (123h) must be setup, as shown in Table 15.

Table 15. Start_config_reg_part1 registers (123h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	start8_sens_uc1	start7_sens_uc1	start6_sens_uc1	start5_sens_uc1	start4_sens_uc1	start3_sens_uc1	start2_sens_uc1	start1_sens_uc1	start8_sens_uc0	start7_sens_uc0	start6_sens_uc0	start5_sens_uc0	start4_sens_uc0	start3_sens_uc0	start2_sens_uc0	start1_sens_uc0
Value	00		0	0	0	0	0	0	0	0	0	0	0	1	0	0

4.1.2.2.2 Entry point registers

Entry point registers define where each microcore starts in each channel. With the new PT2000 IDE, it is possible to specify a label as a start point and the following registers are set automatically. For example the “init0” label is the starting point of Uc0Ch1. If this feature is not used, the following register has to be set manually. In the code example, the entry point of Uc0Ch1 is 0 as the first line executed, and is the first Code RAM line of the channel 1.

Table 16. Uc0_entry_point registers (10Ah) of channel 1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved						entry_point_address									
Reset	000000						0000000000									

The code entry point of Uc1Ch1 is the 31h Code RAM line of the channel 1.

Table 17. Uc1_entry_point registers (10Bh) of channel 1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved						entry_point_address									
Reset	000000						0000110001									

The same configuration must be done for Channel2.

Table 18. Uc0_entry_point registers (12Ah) of channel 2

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved						entry_point_address									
Reset	000000						0000000000									

The code entry point of Uc1Ch1 is the 2Fh Code RAM line of the channel 1.

Table 19. Uc1_entry_point registers (12Bh) of channel 2

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved						entry_point_address									
Reset	000000						0000110001									

4.1.2.3 Diagnosis configuration registers

The high-side and low-side drivers must be directly controlled by the microcores. Consequently, the output_routing fields of the high-side and low-side driver's output configuration register must be set to the value 15. Dead time bits need to be set to avoid cross conduction between high-side and low-side freewheeling.

t_{FWDLY} = Tck x (Dead_time + 1) = 1/6 MHz x (3 + 1) = 668 ns

Table 20. Hsx_output_config registers (1DAh, 1DDh, 1E0h, 1E3h, 1E6h, 1ECh)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	HSx ovr	HSx Vds ref	reserved			dead_time					output_routing					inv
Value	0	0	000			00011					11111					0

Table 21. Lsx_output_config registers (1C2h, 1C5h, 1C8h, 1CBh, 1CEh, 1D1h, 1D4h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	LSx_ovr	Reserved									output_routing					inv
Value	0	000000000									11111					0

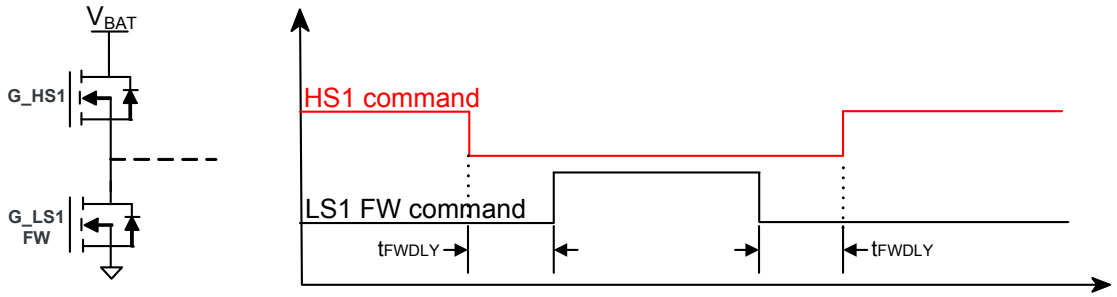

Figure 5. Automatic freewheeling example

4.1.3 Injection banks management algorithm

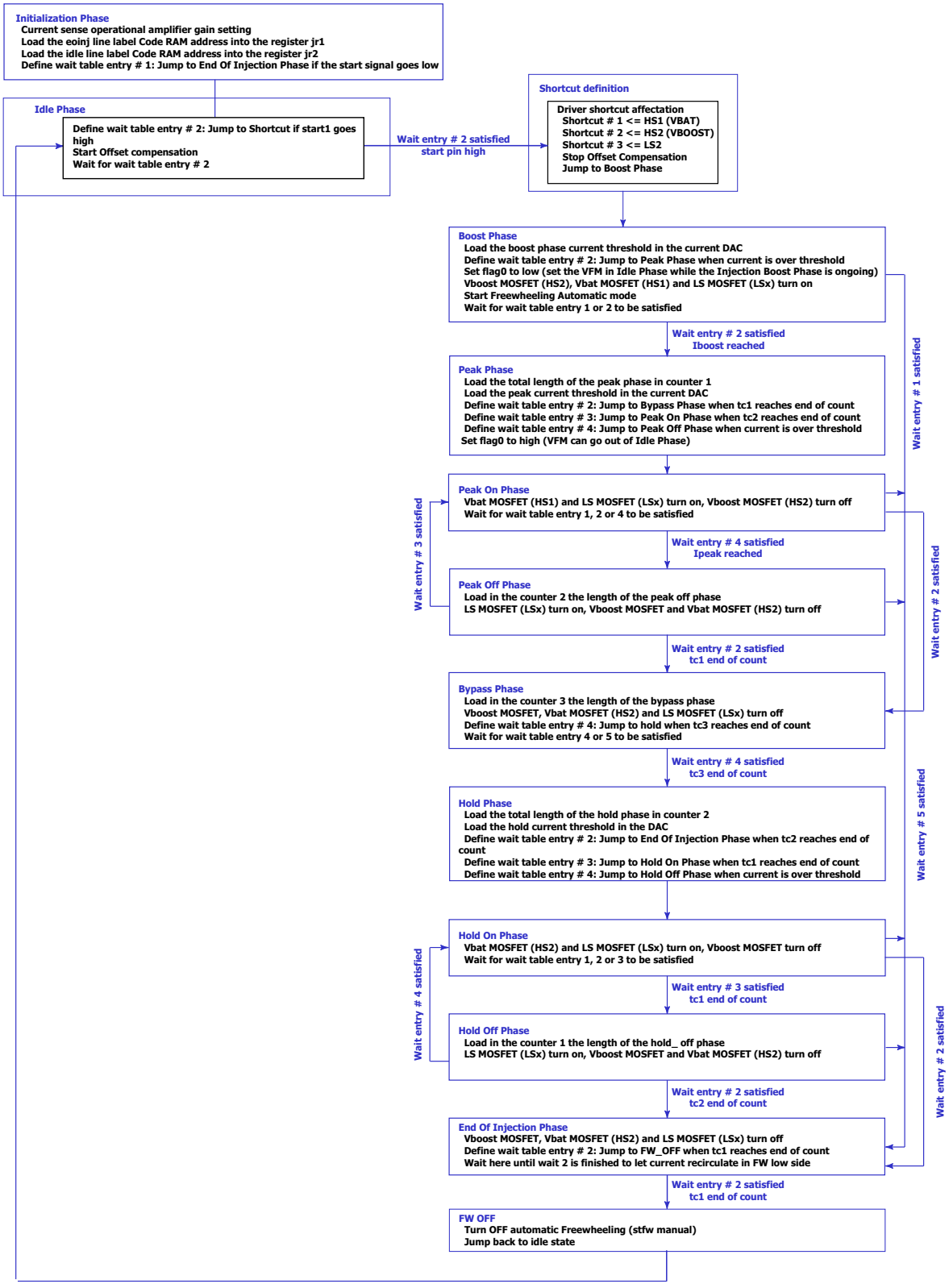

Figure 6. Algorithm example for one injector

Refer to Injection banks management source code

4.2 DC-DC resonant mode management

In Resonant mode, off switching is triggered by the sense current rising above an upper current threshold and the on switching is triggered by the low-side VDS going below 2.5 V. This mode uses a current control loop within a voltage control loop. The voltage control loop is controlled by microcode, V_{BOOST} high and low threshold are defined in the DRAM.

To use this mode on the KITPT2000FRDM3C, the capacitance C_{RES} needs to be populated, it is not the case by default. The code dedicated to the boost converter regulation loop is loaded into the Code RAM 3. This code is executed independently by the microcore Uc0Ch3.

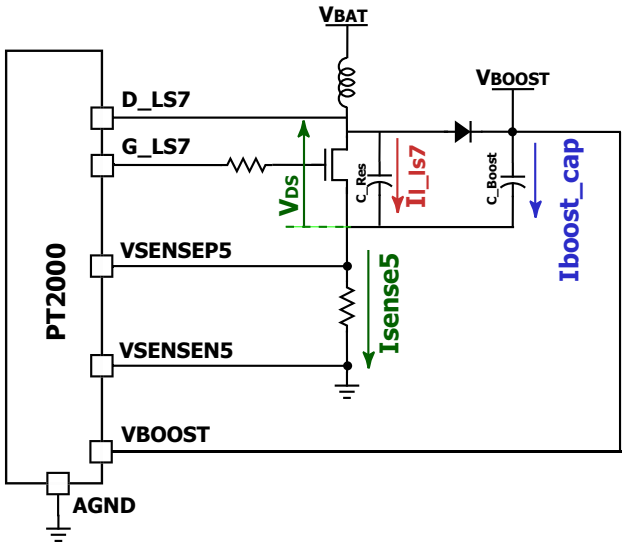

Figure 7. Simplified DC-DC converter topology for resonant mode

At boost startup, the current through the inductor oscillates between a high current threshold and the VDS low-side, this avoid switching loss during the turn OFF. Low-side VDS goes lower than 2.5 V as soon as there is no more current in the inductance, the goal of the C_{RES} is to reduce the switching slew rates to detect when VDS across low-side mosfet goes below 2.5 V with the fast comparator. In case the VDS is not detected (this can be the case if V_{BOOST}-V_{BAT} > V_{BAT}), a timeout is used to turn the low-side ON again.

When this switch is on, the current grows through the sense resistor and the low-side switch. When the switch is open, the current decays through the diode and loads the output capacitor. It increases the voltage until the V_{BOOST} voltage reaches the V_{BOOST_HIGH} threshold. This phase uses the asynchronous mode and the current modulation is managed by an independent circuitry enabled by the microcore.

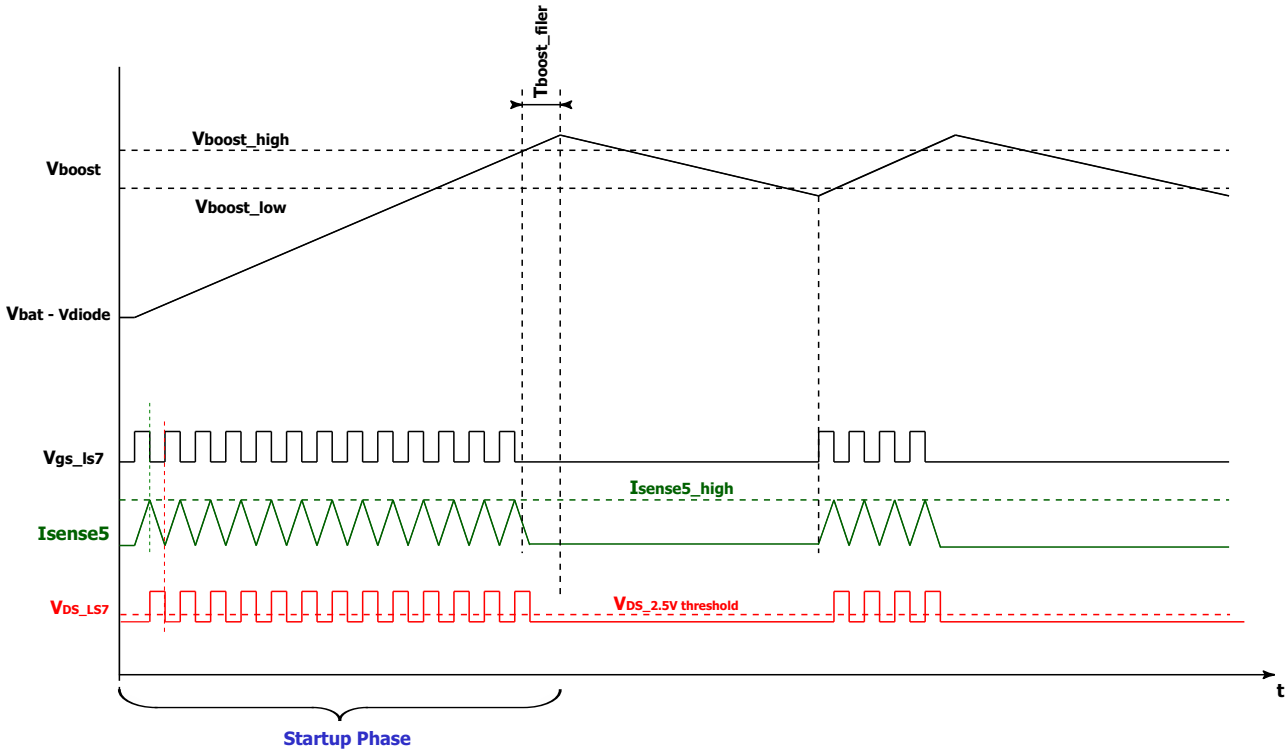

Figure 1. Resonant mode startup sequence

When the V_{BOOST_HIGH} threshold is reached, the synchronous mode is enabled. In this case, the microcore takes the direct control of the low-side switch. The low-side switch is turned off until the boost voltage goes below the V_{BOOST_LOW} threshold.

Each time the V_{BOOST_HIGH} threshold is reached, the V_{BOOST_LOW} threshold is setup, and the synchronous mode is activated after a t_{BOOST_FILTER} filter time required by the voltage comparator circuitry enablement.

Each time the boost voltage falls below the V_{BOOST_LOW} threshold the V_{BOOST_HIGH} threshold is setup, and the asynchronous mode is activated after a t_{BOOST_FILTER} filter time.

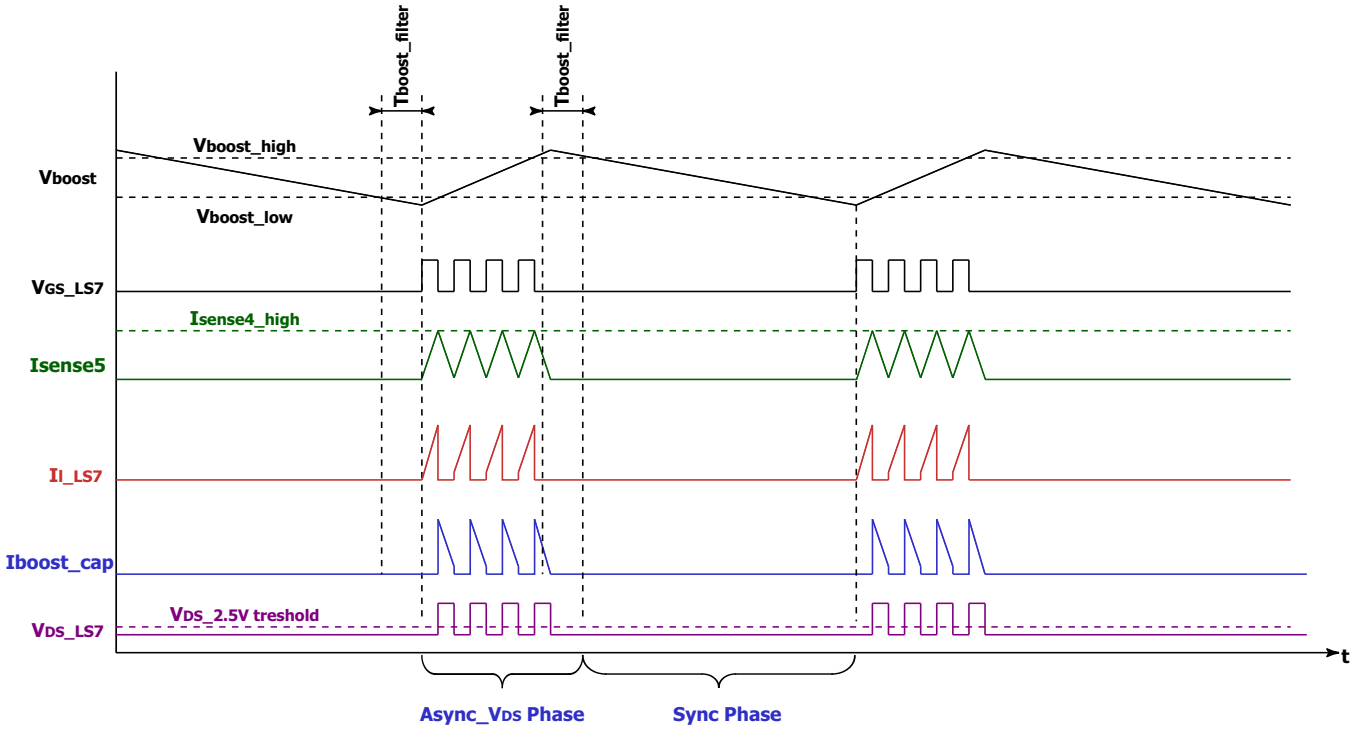

Figure 2. DCDC voltage and current diagram

Table 22. Example of DC-DC converter key parameters

Parameter name	Description	Value
V_{BOOST_HIGH}	V_{BOOST} voltage high threshold	65.31 V
V_{BOOST_LOW}	V_{BOOST} voltage low threshold	64.69 V
I_{SENSE5_LOW}	Low current threshold (only use in case VDS 2.5 V threshold not reached)	0.07 A
I_{SENSE5_HIGH}	High current threshold	3.95 A

In the present case, most of the code branches (jumps) are managed according to the V_{BOOST} voltage and the flag0 state by means of the wait table. The wait table rows are affected as shown in Table 23. and are changed according to the actuation phase.

Table 23. Example of wait table definition for the DCDC.

Phase	Async_Vds phase (dcdc_on)	Sync phase (dcdc_off)	Idle phase
Row 1	If flag0 is low, then jump to Idle phase	If flag0 is low, then jump to Idle phase	-
Row 2	-	If $V_{BOOST} < V_{BOOST\_LOW}$, then jump to Async_Vds Phase	-
Row3	If $V_{BOOST} > V_{BOOST\_HIGH}$, then jump to Sync phase	-	-
Row 4	-	-	-
Row 5	-	-	-

To avoid regulation disturbances, the boost voltage regulation is stopped by the means of the internal flag 0 when an injection phase starts.

4.2.1 DC-DC management registers setup

The PT2000 registers are setup according their default states, unless defined by the following.

4.2.1.1 IO configuration registers

4.2.1.1.1 Output driver access

The Uc0Ch3 must have access to the pre-driver LS7 only.

Table 1. Out_acc_uc0_ch3 register (164h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Acc_uc0_ch3_ls8	Acc_uc0_ch3_ls7	Acc_uc0_ch3_ls6	Acc_uc0_ch3_ls5	Acc_uc0_ch3_ls4	Acc_uc0_ch3_ls3	Acc_uc0_ch3_ls2	Acc_uc0_ch3_ls1	Reserved	Acc_uc0_ch3_hs7	Acc_uc0_ch3_hs6	Acc_uc0_ch3_hs5	Acc_uc0_ch3_hs4	Acc_uc0_ch3_hs3	Acc_uc0_ch3_hs2	Acc_uc0_ch3_hs1
Value	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Resonant mode (“async_vds) is required to set the following register (182h), to set the V_{DS} LS7 monitoring to use the high speed comparator to detect when V_{DS} is lower than 2.5 V, as fast as possible. Also important is to set the bit vds7_en and set the filter time properly to start the DCDC, even if the VDS_threshold is not reached. This a safety mechanism.

Table 2. Vds7_dcdc_config (182h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	dcdcx mode		Reserved			ls7_vds_highspeed_en	Cur_dcdc7_fbk_sel	vds7_to_en	vds7_dcdc_timeout							
Value	00		000			1	0	1	000000011							

LS7 VDS threshold needs to be set to 2.5 V to make the DCDC resonant mode efficient enough.

Table 3. Vds_threshold_ls_Part 2 (170h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Vds thr Ls8				Vds thr Ls7				Vds thr Ls6				Vds thr Ls5			
Reset	0000				0101 (2.5V threshold)				0000				0000			

4.2.1.1.2 Current sense access

The Uc0Ch3 must have access to the current sense feedback # 5L (for the timeout detection, if the 2.5 V VDS threshold is not reached) and 5H.

Table 2. Cur_block_access_3 register (168h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	acc_uc1_ch3_curr6H_6Neg	acc_uc1_ch3_curr6L	acc_uc1_ch3_curr5H_5Neg	acc_uc1_ch3_curr5L	acc_uc1_ch3_curr4	acc_uc1_ch3_curr3	acc_uc1_ch3_curr2	acc_uc1_ch3_curr1	acc_uc0_ch3_curr6H_6Neg	acc_uc0_ch3_curr6L	acc_uc0_ch3_curr5H_5Neg	acc_uc0_ch3_curr5L	acc_uc0_ch3_curr4	acc_uc0_ch3_curr3	acc_uc0_ch3_curr2	acc_uc0_ch3_curr1
Value	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

4.2.1.1.3 Boost DAC access

The Uc0Ch3 must have access to the Boost DAC to set the right threshold Vboost_high and Vboost_low.

Table 4. Boost_dac_access (180h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved										uc1_ch 3 acc	uc0_ch 3 acc	uc1_ch 2 acc	uc0_ch 2 acc	uc1_ch 1 acc	uc0_ch 1 acc
Value	0000000000										0	1	0	0	0	0

4.2.1.1.4 Boost filter time

The t_{BOOST_FILTER} time is defined in the Boost_filter register (181h). This filter time and type can be adjusted to improve the V_{BOOST} voltage stability.

Table 5. Boost_filter (181h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved			filter_type	Boost_fbk_filter											
Value	000			0	00000000111											

4.2.1.2 Diagnosis configuration registers

The low-side driver 7 must be directly controlled by the Uc0Ch3 microcore. Consequently, the output_routing fields of its output configuration register must be set to the value 31.

Table 6. LS7_output_config (1D4h)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	LSx_ovr	Reserved									output_routing					inv
Reset	0	000000000									11111					0

4.2.1.3 Channel 3 configuration registers

Same as for Channel 1-2 entry points of each microcore must be selected for Channel 3.

4.2.1.3.1 Entry point registers

Entry point registers are defining where each microcore starts in each channel. With the new PT2000 IDE, it is possible to specify a label as a starting point and those registers are set automatically. For example, “init0” is the starting point of Uc0Ch3. If this feature is not used, the following register has to be set manually.

In the code example entry point of Uc0Ch3 is 0 as the first line executed, is the first Code RAM line of the channel 1.

Table 3. Uc0_entry_point registers (10Ah) of channel 1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved						entry_point_address									
Reset	000000						0000000000									

Uc1CH3 is not used in our example, but to avoid an issue if the dual sequencer is selected, the code entry point of Uc1Ch3 is set and a dummy code of at least 3 lines is added to the example. This is unusable if the bit dual microcore (140h) is not set to ‘1’.

Table 4. Uc1_entry_point registers (10Bh) of channel 1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Reserved						entry_point_address									
Reset	000000						10001									

4.2.2 DC-DC management algorithm

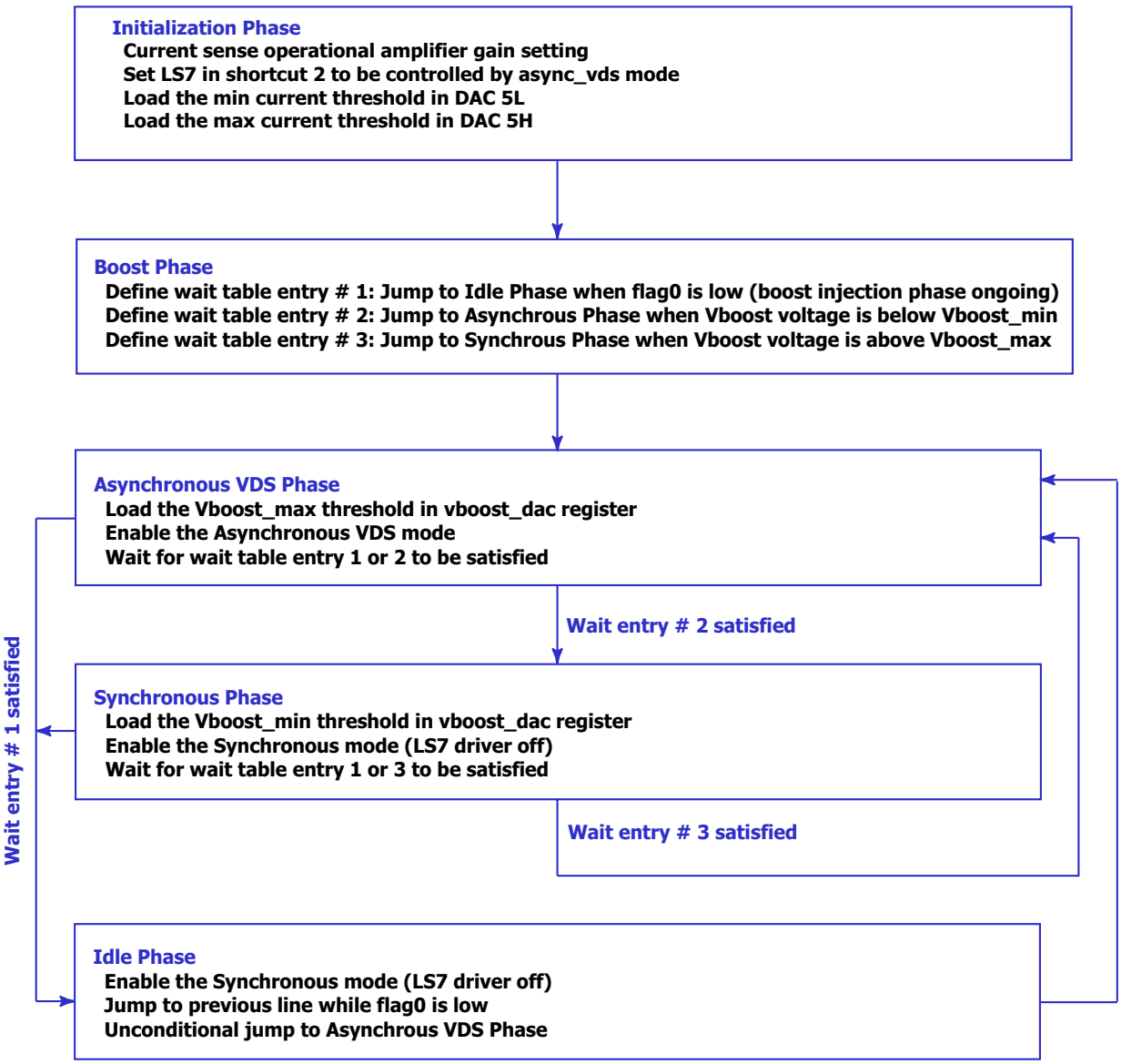

Figure 3. Algorithm example for DCDC

Refer to DC-DC Management Source Code

5 PCB layout recommendations

5.1 Ground connections

The PT2000 exposed pad must be connected to the PCB ground. All the grounds (AGND, DGND, and PGND) must be ‘in star' or considering a unique ground layer, such as to minimize the introduction of offset and noise mainly in the signal return lines.

5.2 Sense resistors connection

The sense resistor’s layout must be considered with special care, to sense the voltage as close as possible to the resistor terminations.

Balanced series resistance, induced by the layout, between the sense resistor positive termination to the VSENSEPx and the sense resistor negative termination to the VSENSENx pin is recommended. The balance can be achieved by implementing similar line lengths.

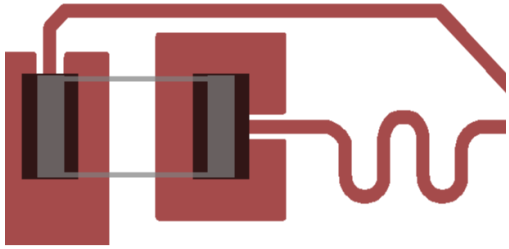

Figure 4. Example of force and sense connection layout

It is highly recommended to place the sense resistor as close as possible to its corresponding low-side MOSFET transistor.

6 Application source code

Download the full project, including microcode, plus register and DRAM settings, at this location
http://www.nxp.com/files/analog/doc/app_note/AN5187SW.zip

6.1 Injection banks management source code

```
*************************************************************************
*                                          Copyright (c) NXP 2014       *
* File Name: MC33PT2000_VFM_65V.dfi                                     *
* Current Revision: 1.0                                                 *
* Purpose: MC33PT2000 example - 1 Bank                                  *
* Description:  MC33PT2000 Channel 1 controls INJ1 and INJ2             *
*                                                                       *
* REV  AUTHOR    DATE        DESCRIPTION OF CHANGE                      *
* ---  ------    --------    ---------------------                      *
* 1.0  b16868    2015/03/25     - initial coding                        *
*                                                                       *
*************************************************************************

*************************************************************************
* NXP reserves  the right to make  changes without  further notice to any *
* product herein to improve  reliability, function,  or design.  NXP does *
* not  assume  any  liability  arising  out of  the application  or use  of any *
* product, circuit,  or software described herein;  neither does  it convey any *
* license under its patent rights nor the rights of others.  NXP products *
* are not designed, intended,  or authorized for use as  components  in systems *
* intended for  surgical implant into the body,  or other applications intended *
* to support life,  or for any other  application  in which the failure  of the *
* NXP product could create a situation where  personal injury or death may *
* occur.  Should Buyer purchase or use NXP products for any such intended *
* or unauthorized  application,  Buyer shall indemnify and  hold  NXP and *
* its officers, employees, subsidiaries, affiliates,  and distributors harmless *
* against all claims costs, damages, and expenses, and reasonable attorney  fees*
* arising  out of,  directly or indirectly,  any claim  of personal  injury  or *
* death  associated  with  such unintended  or unauthorized use,  even if  such *
* claim  alleges  that  NXP   was  negligent   regarding   the  design  or *
* manufacture  of the part.  NXP and the  NXP logo  are registered *
* trademarks of NXP Ltd.                                                *
*************************************************************************

* This microcore uc0 will control BANK1
* High SIde Vbat = hs1
* High SIde Vboost = hs2
* Low side Freewheeling = ls1
* Low side = ls2
* current sense = cur1
#include "dram1.def";

*****************************************************************************************************
*                                        INIT PHASE
*****************************************************************************************************

* ### Initialization phase ###
init0:      stgn gain8.68 sssc; * Set the gain of the opamp of the current measure block 1
            ldjr1 eoinj0; * Load the eoinj line label Code RAM address into the register jr1
            ldjr2 idle0; * Load the idle line label Code RAM address into the register jr2
            cwef jr1 _start row1; * If the start signal goes low, go to eoinj phase
            sto ls1 off;

*****************************************************************************************************
*                                        IDLE PHASE                                                 *
*****************************************************************************************************
* ### Idle phase- the uPC loops here until start signal is present ###
idle0:      stoc on sssc;                          * Turn ON offset compensation
```

```
            cwer CheckStart start row2;             * Define entry table for high start pin
WaitLoop:   wait row2;                              * uPC is stuck here for almost the whole idle time
CheckStart: joslr inj1_start start1;                * Jump to inj1 if start 1 is high
            jmpr WaitLoop;

*****************************************************************************************************
*                                        SHORTCUT DEFINITION                                        *
*****************************************************************************************************

* ### Shortcuts definition per the injector to be actuated ###
inj1_start: dfsct hs1 hs2 ls2;                      * Set the 3 shortcuts: VBAT, VBOOST, LS2
            stoc off sssc;                          * Disable Offset Compensation


*****************************************************************************************************
*                                             BOOST PHASE                                           *
*****************************************************************************************************

* ### Launch phase enable boost ###
boost0:     load Iboost dac_sssc _ofs;              * Load the boost phase current threshold in the current DAC
            cwer peak0 cur1 row2;                   * Jump to peak phase when current is over threshold
            stf low b0;                             * set flag0 low to force the DC-DC converter in idle mode
            stos on on on;                         * Turn VBAT off, BOOST on, LS on
            stfw auto;                              * LS1 is used as the freewheeling of hs1
            stf high b1;                            * Test purpose
            wait row12;                             * Wait for one of the previously defined conditions


*****************************************************************************************************
*                                             PEAK PHASE                                            *
*****************************************************************************************************
* ### Peak phase continue on Vbat ###
peak0:      ldcd rst _ofs keep keep Tpeak_tot c1;   * Load the length of the total peak phase in counter 1
            load Ipeak dac_sssc _ofs;               * Load the peak current threshold in the current DAC
            cwer bypass0 tc1 row2;                  * Jump to bypass phase when tc1 reaches end of count
            cwer peak_on0 tc2 row3;                 * Jump to peak_on when tc2 reaches end of count
            cwer peak_off0 cur1 row4;               * Jump to peak_off when current is over threshold
            stf high b0;                            * set flag0 high to release the DC-DC converter idle mode

peak_on0:   stos on off on;                         * Turn VBAT on, BOOST off, LS on
            wait row124;                            * Wait for one of the previously defined conditions

peak_off0:  ldcd rst ofs keep keep Tpeak_off c2;    * Load in the counter 2 the length of the peak_off phase
            stos off off on;                        * Turn VBAT off, BOOST off, LS on
            wait row123;

*****************************************************************************************************
*                                             BYPASS PHASE                                          *
*****************************************************************************************************

* ### Bypass phase ###
bypass0:    ldcd rst ofs keep keep Tbypass c3;      * Load in the counter 3 the length of the off_phase phase
            stos off off off;                       * Turn VBAT off, BOOST off, LS off
            cwer hold0 tc3 row4;                    * Jump to hold when tc3 reaches end of count
            wait row14;                             * Wait for one of the previously defined conditions

*****************************************************************************************************
*                                             HOLD PHASE                                            *
*****************************************************************************************************

* ### Hold phase on Vbat ###
hold0:      ldcd rst _ofs keep keep Thold_tot c1;   * Load the length of the total hold phase in counter 2
            load Ihold dac_sssc _ofs;               * Load the hold current threshold in the DAC
            cwer eoinj0 tc1 row2;                   * Jump to eoinj phase when tc1 reaches end of count
            cwer hold_on0 tc2 row3;                 * Jump to hold_on when tc2 reaches end of count
            cwer hold_off0 cur1 row4;               * Jump to hold_off when current is over threshold
```

```
hold_off0:  ldcd rst _ofs keep keep Thold_off c2;   * Load the length of the hold_off phase in counter 1
            stos off off on;                        * Turn VBAT off, BOOST off, LS on
            wait row123;

hold_on0:   stos on off on;                         * Turn VBAT on, BOOST off, LS on
            wait row124;                            * Wait for one of the previously defined conditions

*****************************************************************************************************
*                                    END OF INJECTION PHASE                                         *
*****************************************************************************************************

* ### End of injection phase ###
eoinj0:     stos off off off;                       * Turn VBAT off, BOOST off, LS off
            stf high b0;                            * set flag0 to high to release the DC-DC converter idle mode
            ldcd rst _ofs keep keep FWtimeOFF c1;   * Load the length of the time when FW low side will stay ON
            cwer FW_OFF tc1 row2;                   * Jump to eoinj phase when tc1 reaches end of count
            wait row2;                              * Wait until FW phase is complete
FW_OFF:     stfw manual;                            * Turn OFF FW low side, go back to manual mode
            jmpf jr2;                               * Jump back to idle phase

* ### End of Channel 1 - uCore0 code ###


*#####################################################################################################################
#################
* This microcore uc1 will control BANK2
* High SIde Vbat = hs3
* High SIde Vboost = hs4
* Low side Freewheeling = ls3
* Low side = ls4
* current sense = cur2

*#####################################################################################################################
#################


*****************************************************************************************************
*                                        INIT PHASE
*****************************************************************************************************
* ### Initialization phase ###
init1:      stgn gain8.68 sssc;                     * Set the gain of the opamp of the current measure block 1
           ldjr1 eoinj1;                          * Load the eoinj line label Code RAM address into the register jr1
            ldjr2 idle1;                           * Load the idle line label Code RAM address into the register jr2
            cwef jr1 _start row1;                   * If the start signal goes low, go to eoinj phase
            sto ls3 off;

*****************************************************************************************************
*                                        IDLE PHASE                                                 *
*****************************************************************************************************
* ### Idle phase- the uPC loops here until start signal is present ###
idle1:      stoc on sssc;                           * Turn ON offset compensation
            cwer CheckStart1 start row2;             * Define entry table for high start pin
WaitLoop1:   wait row2;                             * uPC is stuck here for almost the whole idle time
CheckStart1: joslr inj2_start start2;               * Jump to inj2 if start 2 is high
            jmpr WaitLoop1;

*****************************************************************************************************
*                                      SHORTCUT DEFINITION                                          *
*****************************************************************************************************

* ### Shortcuts definition per the injector to be actuated ###
inj2_start: dfsct hs3 hs4 ls4;                      * Set the 3 shortcuts: VBAT, VBOOST, LS4
            stoc off sssc;                          * Stop offset compensation
```

```
*****************************************************************************************************
*                                          BOOST PHASE                                              *
*****************************************************************************************************
* ### Launch phase enable boost ###
boost1:     load Iboost dac_sssc _ofs;             * Load the boost phase current threshold in the current DAC
            cwer peak1 cur2 row2;                  * Jump to peak phase when current is over threshold
            stf low b0;                            * set flag0 low to force the DC-DC converter in idle mode
            stos on on on;                        * Turn VBAT off, BOOST on, LS on
            stfw auto;                             * LS1 is used as the freewheeling of hs3
            stf high b2;
            wait row12;                            * Wait for one of the previously defined conditions

*****************************************************************************************************
*                                          PEAK PHASE                                               *
*****************************************************************************************************
* ### Peak phase continue on Vbat ###
peak1:      ldcd rst _ofs keep keep Tpeak_tot c1;  * Load the length of the total peak phase in counter 1
            load Ipeak dac_sssc _ofs;              * Load the peak current threshold in the current DAC
            cwer bypass1 tc1 row2;                 * Jump to bypass phase when tc1 reaches end of count
            cwer peak_on1 tc2 row3;                * Jump to peak_on when tc2 reaches end of count
            cwer peak_off1 cur2 row4;              * Jump to peak_off when current is over threshold
            stf high b0;                           * set flag0 high to release the DC-DC converter idle mode

peak_off1:  ldcd rst ofs keep keep Tpeak_off c2;   * Load in the counter 2 the length of the peak_off phase
            stos off off on;                       * Turn VBAT off, BOOST off, LS on
            wait row123;

peak_on1:   stos on on on;                        * Turn VBAT on, BOOST off, LS on
            wait row124;                           * Wait for one of the previously defined conditions


*****************************************************************************************************
*                                          BYPASS PHASE                                             *
*****************************************************************************************************


* ### Bypass phase ###
bypass1:    ldcd rst ofs keep keep Tbypass c3;     * Load in the counter 3 the length of the off_phase phase
            stos off off off;                      * Turn VBAT off, BOOST off, LS off
            cwer hold1 tc3 row4;                   * Jump to hold when tc3 reaches end of count
            wait row14;                            * Wait for one of the previously defined conditions

*****************************************************************************************************
*                                          HOLD PHASE                                               *
*****************************************************************************************************

* ### Hold phase on Vbat ###
hold1:      ldcd rst _ofs keep keep Thold_tot c1;  * Load the length of the total hold phase in counter 2
            load Ihold dac_sssc _ofs;              * Load the hold current threshold in the DAC
            cwer eoinj1 tc1 row2;                  * Jump to eoinj phase when tc1 reaches end of count
            cwer hold_on1 tc2 row3;                * Jump to hold_on when tc2 reaches end of count
            cwer hold_off1 cur2 row4;              * Jump to hold_off when current is over threshold

hold_off1:  ldcd rst _ofs keep keep Thold_off c2;  * Load the length of the hold_off phase in counter 1
            stos off off on;                       * Turn VBAT off, BOOST off, LS on
            wait row123;

hold_on1:   stos on off on;                       * Turn VBAT on, BOOST off, LS on
            wait row124;                           * Wait for one of the previously defined conditions


*****************************************************************************************************
*                                    END OF INJECTION PHASE                                         *
*****************************************************************************************************
```

```
* ### End of injection phase ###
eoinj1:     stos off off off;                       * Turn VBAT off, BOOST off, LS off
            stf high b0;                            * set flag0 to high to release the DC-DC converter idle mode
            ldcd rst _ofs keep keep FWtimeOFF c1;   * Load the length of the time when FW low side will stay ON
            cwer FW_OFF1 tc1 row2;                   * Jump to eoinj phase when tc1 reaches end of count
            wait row2;                              * Wait until FW phase is complete
FW_OFF1:    stfw manual;                            * Turn OFF FW low side, go back to manual mode
            jmpf jr2;                               * Jump back to idle phase

* ### End of Channel 1 - uCore1 code ###
```

6.2 DC-DC Management Source Code

```
*####                    DCDC mode used in this case is resonant                              ####*
*#### Some changes on the hardware needs to be done to go to resonant mode or freewheeling   ####*

 #include "dram3.def";


* ### Initialization phase ###
init0:      sl56dac dac5;                              * DAC5 is used for DCDC
           stgn gain19.25 sssc;                       * Set the gain of the opamp of the current measure block 56
           dfsct undef ls7 undef;                    * NEW on the PT2000 only shortcut 2 is sensitve to async_vds
instruction
            stos keep off keep;
           load Iboost_L dac_sssc _ofs;              * Load Isense5L just used for the timeout detection in case
2.5V VDS not reached
            load Iboost_H dac56h56n _ofs;              * Load Isense56_high current threshold in DAC 56H
            stdm null;                                 * Set the boost voltage DAC access mode
          cwer dcdc_idle _f0 row1;                   * Wait table entry for Vboost under Vboost_low threshold condition
          cwer dcdc_on _vb row2;                     * Wait table entry for Vboost under Vboost_low threshold condition
          cwer dcdc_off vb row3;                     * Wait table entry for Vboost over Vboost_high threshold condition

* ### Asynchronous phase ###
dcdc_on:    load Vboost_H dac56h56n _ofs;              * Load the upper Vboost threshold in vboost_dac register
            stdcctl async_vds;                         * Enable asynchronous mode
            wait row13;                                * Wait for one of the previously defined conditions

* ### Synchronous phase ###
dcdc_off:   load Vboost_L dac56h56n _ofs;              * Load the upper Vboost threshold in vboost_dac register
            stdcctl sync;                              * Enable synchronous mode
            wait row12;                                * Wait for one of the previously defined conditions

* ### Idle phase ###
dcdc_idle:  stdcctl sync;                              * Enable synchronous mode
            stos keep off keep;
            jocr dcdc_idle _f0;                        * jump to previous line while flag 0 is low
            jmpr dcdc_on;                              * force the DC-DC converter on when flag 0 goes high

* ### End of Channel 3 - uCore0 code ###
```

7 References

Following are URLs where you can obtain information on NXP products and application solutions:

Document Number and Description		URL
PT2000	Data Sheet	http://www.nxp.com/files/analog/doc/data_sheet/MC33PT2000.pdf
KITPT2000FRDM6C	User Guide	http://cache.nxp.com/files/analog/doc/user_guide/KTPT2000FRDM6CUG.pdf
PT2000SWUG	Programming Guide	http://cache.nxp.com/files/analog/doc/user_guide/PT2000SWUG.pdf
PT2000IDEUG	Developer Studio	http://cache.nxp.com/files/analog/doc/user_guide/PT2000-IDEUG.pdf
Support Pages		**URL**
NXP.com		http://www.nxp.com
Product Summary Page		http://www.nxp.com/webapp/sps/site/prod_summary.jsp?code=PT2000
Analog Home Page		http://www.nxp.com/analog

8 Revision history

Revision	Date	Description
1.0	9/2015	• Initial release
	7/2016	• Updated to NXP document form and style

How to Reach Us:

Home Page:
NXP.com

Web Support:
http://www.nxp.com/support

Document Number: AN5187
Rev. 1.0
7/2016