Temperature Monitoring and Fan Control with Platform Manager 2

September 2018 **Technical Note FPGA-TN-02080**

Introduction

Platform Manager™ 2 devices are fast-reacting, programmable logic based hardware management controllers. Platform Manager 2 devices are an integrated solution combining analog sense and control elements with scalable programmable logic resources. This integrated approach allows Platform Manager 2 designs to address Power Management (Power Sequencing, Voltage Monitoring, Trimming and Margining), Thermal Management (Temperature Monitoring, Fan Control, Power Control), and Control Plane functions (System Configuration, I/O Expansion, and others). This technical note focuses on the Temperature Monitoring and Fan Control features, and the process for integrating these features into the overall hardware management control.

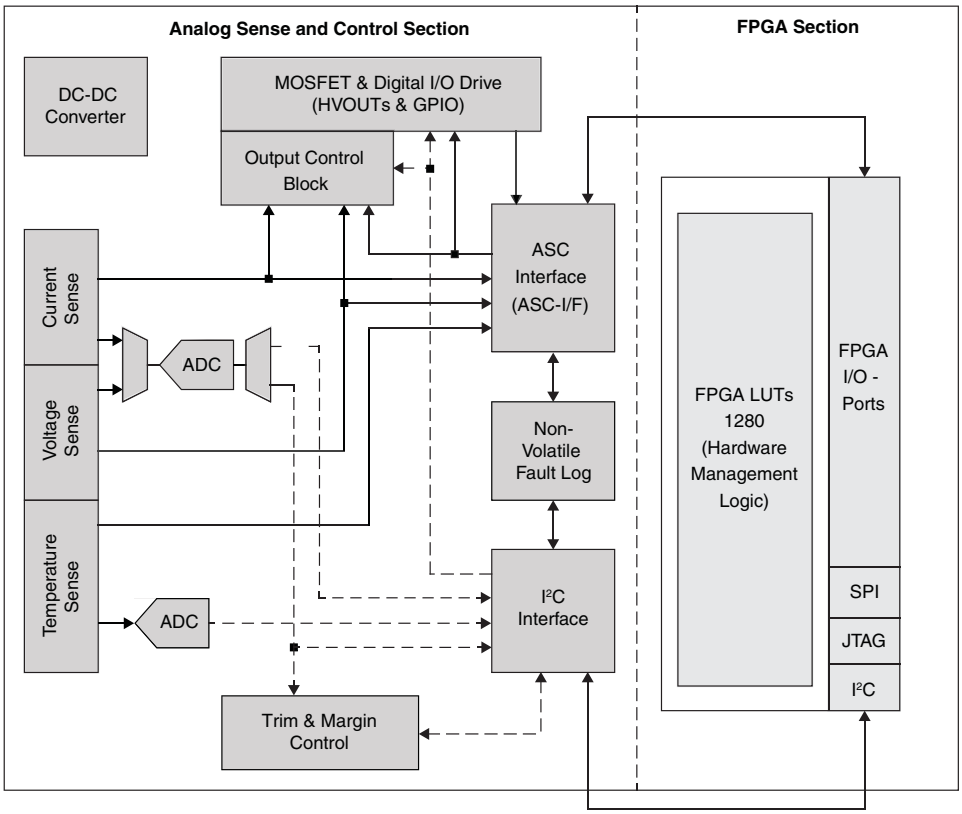
The information in this technical note applies to all Platform Manager 2 designs. Platform Manager 2 designs can be built using several combinations of Lattice devices as listed in Table 1. Throughout the remainder of this document, the use of ASC refers to either the LPTM21L or the L-ASC10 hardware management expander.

Table 1. Lattice Platform Manager 2 Design Options

Central Hardware Manager	Hardware Management Expander	Number of Expanders Supported[1, 2]
LPTM21	LPTM21L or L-ASC10	0 – 3
LPTM21L	LPTM21L or L-ASC10	0 – 3
MachXO2™	LPTM21L or L-ASC10	1 – 8
MachXO3™	LPTM21L or L-ASC10	1 – 8
ECP5™	LPTM21L or L-ASC10	1 – 8

1. Platform Manager 2 designs with 6 hardware expanders are best supported with MachXO2 and MachXO3 devices of 2k LUTs or larger.
2. Platform Manager 2 designs with 8 hardware expanders are best supported with MachXO2 and MachXO3 devices of 4k LUTs or larger.

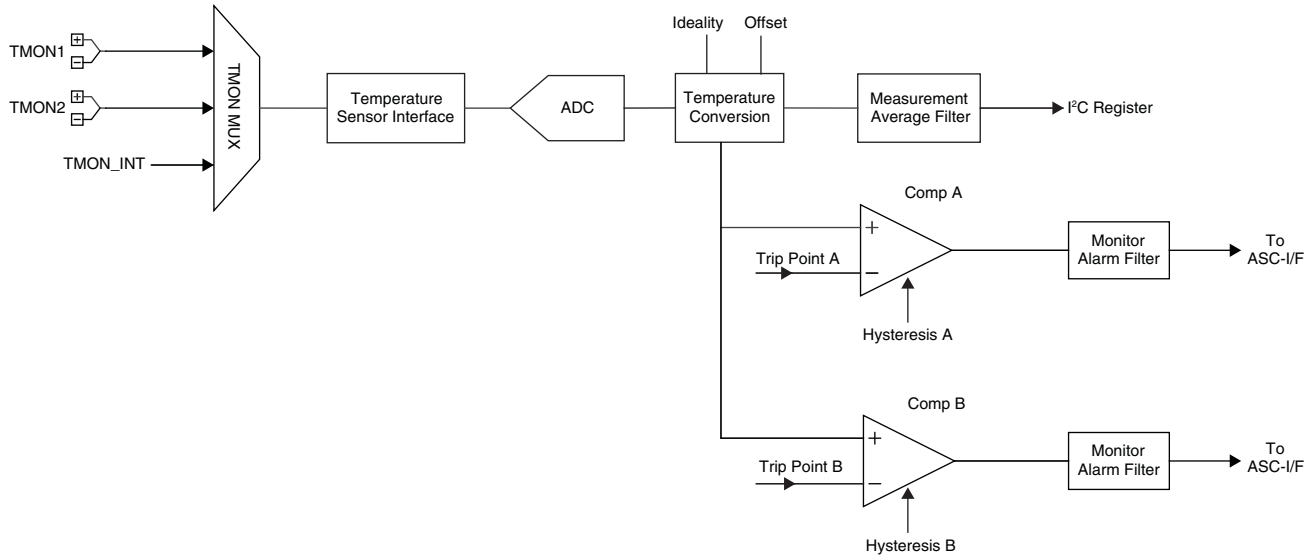
Figure 1. Platform Manager 2 Block Diagram



Overview

Each LPTM21, LPTM21L, or L-ASC10 device includes three programmable temperature monitor circuits (shown in Figure 2). These circuits can be used to monitor the temperature of CPU/ASIC/SOC devices using their integrated temperature sensing diodes or the temperature of the PCB using discrete PNP or NPN transistors. The programmable circuit will generate alarm signals based on user configured over or under temperature thresholds. A variety of compensation and filtering parameters are also available for optimizing the temperature monitoring performance for a specific application.

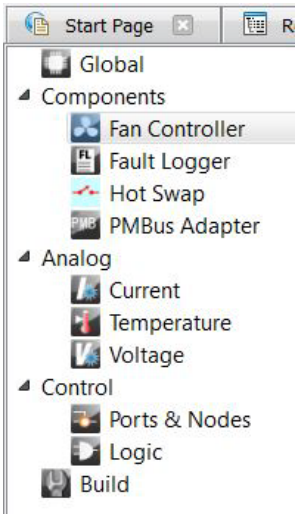
Figure 2. Temperature Monitor Circuit



The monitor alarm signals from the temperature monitor circuit are provided to the FPGA logic in the Platform Manager 2. The FPGA logic implements the hardware control algorithms for the system, including the fan controller IP component. The fan controller is designed to work with a variety of fans and control circuits.

The Platform Designer tool (a part of Lattice Diamond software) is used to configure both the temperature monitor circuit (in the Analog design section) and the fan controller IP (in the Component design section). Figure 3 shows the selectable views in Platform Designer.

Figure 3. Platform Designer Software Tool Views



Temperature Monitor Features

Platform Designer provides a spreadsheet interface for updating the design parameters associated with the temperature monitor circuit. The parameters are shown in two parts (Figure 4 and Figure 7) and are listed with a short explanation below. See the References section for documents with additional details on each parameter.

Figure 4. Temperature Monitor Parameters - Part 1

	ASC Device	Pin	Schematic Net Name	Logical Name	Monitoring Type	Trip Point Selection (C : -64 ~ 155)	Hysteresis (C : 0 ~ 63)	Monitor Alarm Filter (Depth)
1	ALL	TMONs			OT / UT	0	1	1
2	ASC0	TMON1	ASC0_TMON1	ASC0_TMON1_A	OT	0	1	1
3	ASC0	TMON1	ASC0_TMON1	ASC0_TMON1_B	UT	0	1	1
4	ASC0	TMON2	ASC0_TMON2	ASC0_TMON2_A	OT	0	1	1
5	ASC0	TMON2	ASC0_TMON2	ASC0_TMON2_B	UT	0	1	1
6	ASC0	TMON_Int	ASC0_TMON_Int	ASC0_TMON_Int_A	OT	0	1	1
7	ASC0	TMON_Int	ASC0_TMON_Int	ASC0_TMON_Int_B	UT	0	1	1

Schematic Net Name – User defined name which normally corresponds to the signal net name on the user schematic. Changing this parameter from the default value is not required for the application, it is provided for customer reference only.

Logical Name (A and B) – User defined name typically used to convey the alarm signal meaning for in the hardware management logic design. As an example, if ASC0_TMON1_A alarm monitor is configured as an overtemperature warning level for an ASIC in your system, you may update the Logical Name to MyASIC_OT_WARN.

Monitoring Type (A and B) – The monitoring type can be set to over temperature (OT) or under temperature (UT), and is set independently for the A and B alarm signals. The monitoring type does not affect the alarm signal polarity, the alarm signals are always 1 when the measured temperature is above the selected trip point, and 0 when below the trip point. The monitoring type affects the programmable hysteresis behavior. See Figure 5 and Figure 6 for more details.

Trip Point Selection (A and B) – The programmable trip points are used to set the alarm monitor behavior for the A and B temperature comparators. Values above the trip point will set the alarm signal to 1, and values below the trip point will set the alarm signal to 0. Allowed values are between -64C and 155C. The alarm signal behavior is also defined by the hysteresis and monitor alarm filter settings. See Figure 5 and Figure 6 for more details.

Hysteresis (A and B) – The programmable hysteresis, along with the monitoring type and trip point, define the set/reset behavior of the alarm monitor comparators. Allowed values for the hysteresis are between 0C and 63C. See Figure 5 and Figure 6 for more details.

Figure 5. Monitor Alarm Signal Hysteresis - Overtemperature (OT) Setting

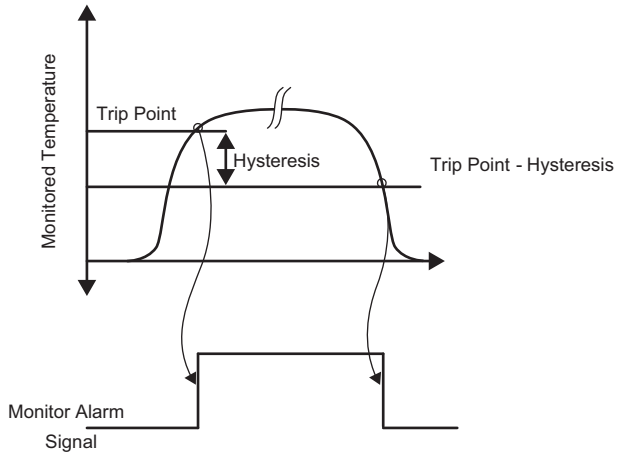
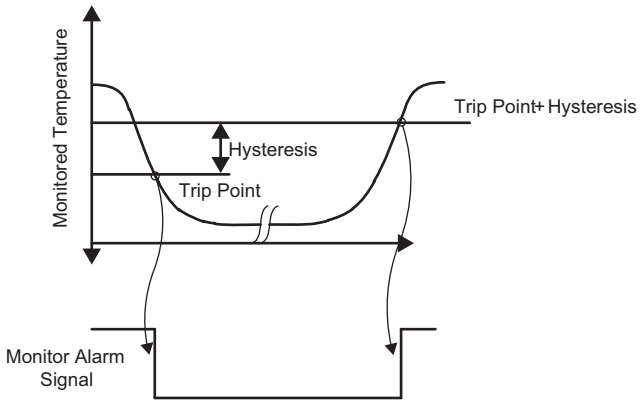


Figure 6. Monitor Alarm Signal Hysteresis - Undertemperature (UT) Setting



Monitor Alarm Filter (A and B) – The monitor alarm signals can be passed to a programmable filter before being output to the ASC-I/F and the FPGA logic. The filter depth is programmable between 1 and 16. The filter circuit increments a counter each time the monitored temperature is above the programmable threshold, and decrements whenever the monitor temperature is below the programmed threshold. When the counter exceeds the programmable depth, the alarm signal is asserted.

Figure 7. Temperature Monitor Parameters - Part 2

	ASC Device	Pin	Measurement Averaging (Filter Coefficient)	Offset (C : -64 ~ 63.75)	Short Fault Measurement Reading (C)	Ideality Factor (0.900 ~ 2.000)	Sensor Configuration
1	ALL	TMONs	1	0	255.75	0.9999	Disabled
2	ASC0	TMON1	1	0	255.75	0.9999	Disabled
3	ASC0	TMON1	1	0	255.75	0.9999	Disabled
4	ASC0	TMON2	1	0	255.75	0.9999	Disabled
5	ASC0	TMON2	1	0	255.75	0.9999	Disabled
6	ASC0	TMON_Int	1	0	255.75	0.9999	
7	ASC0	TMON_Int	1	0	255.75	0.9999	

Measurement Averaging (Filter Coefficient) – The temperature monitor circuit supports a programmable binary exponential averaging filter, which is only applied to the I^2C temperature measurement output (not the temperature monitor alarm outputs).

Offset – Each temperature monitor circuit supports a programmable offset. The offset is applied to the measured temperature prior to comparison to the alarm thresholds. The offset can be used to calibrate out systematic temperature measurement errors in the application.

Short Fault Measurement Reading – The temperature monitor circuits support detection of both open and short faults for remote diode failures. The short fault measurement reading setting changes the behavior of I^2C measurement readings for the open and fault conditions.

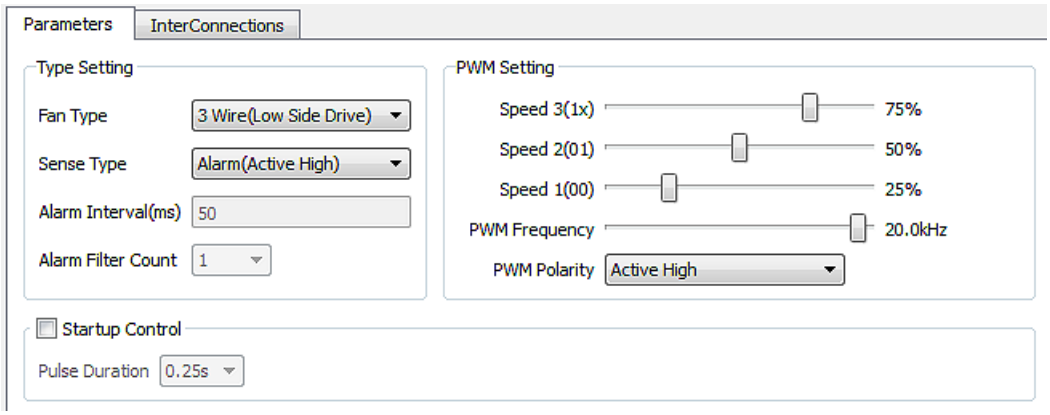
Ideality Factor – The temperature monitor circuit supports a programmable ideality factor. This programmable parameter is used to compensate for non-ideality of the remote sensing diode. The ideality is often given in the transistor or IC data sheet. For a description of how to calculate the ideality for a given device, see the ASC data sheet.

Sensor Configuration – The temperature monitor circuit can be programmed to work with beta-compensated PNP diodes (preferred configuration), differential NPN or PNP diodes, or single-ended diodes (not recommended). The temperature monitor performance has been characterized using the MMBT3906LT3G from ON Semiconductor and the MMBT3906 from Fairchild Semiconductor. The beta-compensated PNP configuration was used for characterization. See the ASC data sheet for more details on the sensor configuration.

Fan Controller Features

Platform Designer provides a GUI interface for updating the design parameters and connections associated with the fan controller component (shown in Figure 8). The parameters and interconnections are listed with a short explanation below.

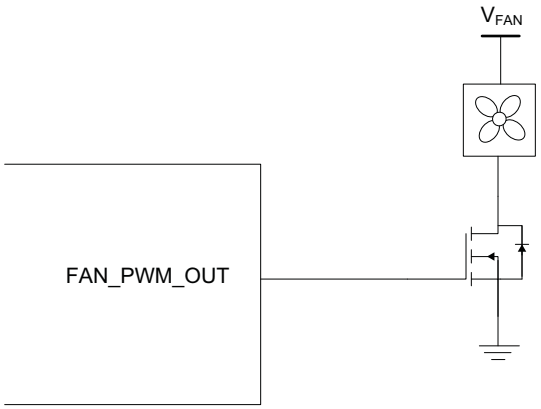
Figure 8. Fan Controller Parameters



Fan Type – There are 4 settings available for the fan type. The fan type settings are denoted by the number of wires (leads) connected to the fan and the drive circuit:

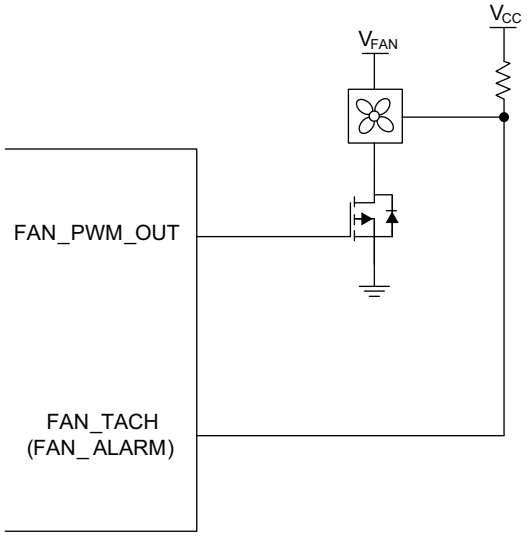
- **2-Wire** – A 2-wire fan has connections for ground and supply. The supply voltage differs by fan. 2-wire fans provide no sensing feedback and require an external circuit for speed control. 2-wire fans are normally controlled by switching a low-side circuit using an FPGA output, as shown in Figure 9, although high side control is also possible.

Figure 9. 2-Wire Fan Connection Circuit



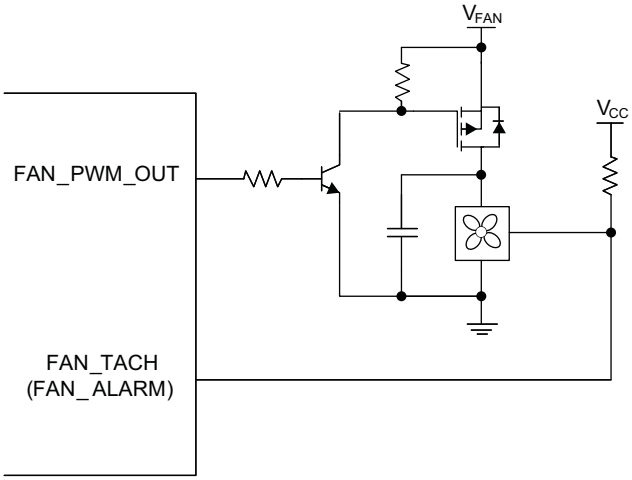
- **3-Wire (Low Side Drive)** – A 3-wire fan has connections for ground, supply, and a tachometer or alarm output. The supply voltage differs by fan. The sensing output from the fan (tachometer or alarm) is monitored by an FPGA input. The sense output is usually an open-drain or open-collector output circuit. The fan controller component will alarm the hardware management control logic when the fan is underspeed or stuck, depending on the alarm settings. 3-wire (low-side drive) fans are controlled by switching a low-side circuit using an FPGA output, as shown in Figure 10. The 3-Wire (Low Side Drive) fan controller implements pulse stretching, to ensure accuracy in underspeed/stuck fan detection. See the section on Pulse Stretching for more details.

Figure 10. 3-Wire (Low Side Drive) Fan Connection Circuit



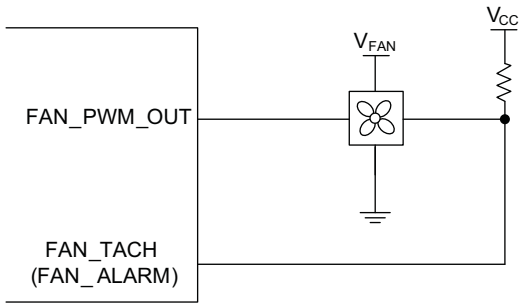
- **3-Wire (High Side Drive)** – The 3-wire high side drive fan type has the same connections as the low side type. In the high side case, the fan is controlled by switching a high-side circuit using an FPGA output. An additional capacitor must be inserted in parallel to the fan between supply and ground, as shown in Figure 11. The 3-wire high-side drive fan type control does not implement pulse stretching; this requirement is removed by the presence of the additional capacitor. For details on sizing the capacitor, see the section on Hold-Up Capacitor Sizing.

Figure 11. 3-Wire (High-Side Drive) Fan Connection Circuit



- **4-Wire** – The 4-wire fan type has connections for ground, supply, speed/PWM, and tachometer or alarm output. The 4-wire fan has an internal switching circuit, which switches the fan based on the speed/PWM input. The typical connection is shown in Figure 12. The 4-wire fan does not require an additional capacitor or pulse stretching, since there is no added control circuit which interrupts the supply to the tachometer or lock detect circuit.

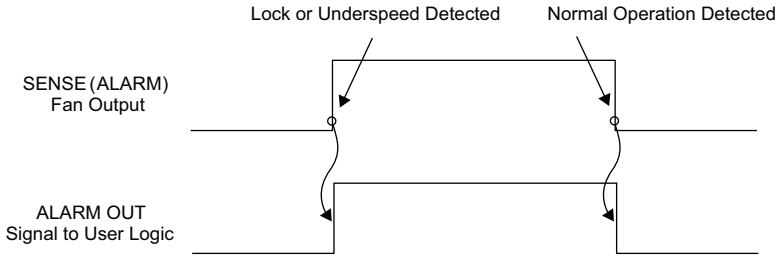
Figure 12. 4-Wire Fan Connection Circuit



Sense Type – The Sense Type parameter is applicable for fans with a sense output (3-wire or 4-wire types).

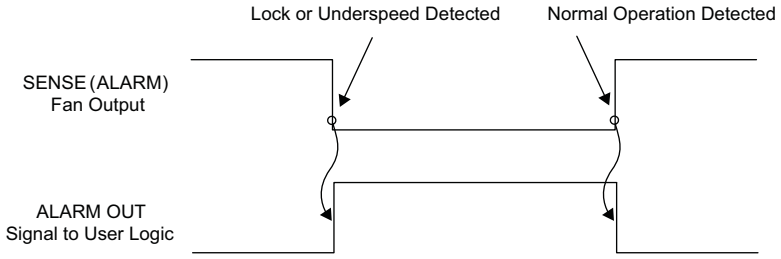
- **Alarm (Active High)** – Fans with an active high sense output follow the behavior shown in Figure 13. If the fan detects an underspeed or lock condition, the alarm output is driven high (The definition of an underspeed condition can be found in the fan data sheet). The fan drives the output low when the underspeed or lock condition is no longer detected. The fan controller component will immediately assert the ALARM OUT signal to the user logic when the sense input at the FPGA is driven high (See the interconnections section for more details).

Figure 13. Alarm (Active High) Sense Type



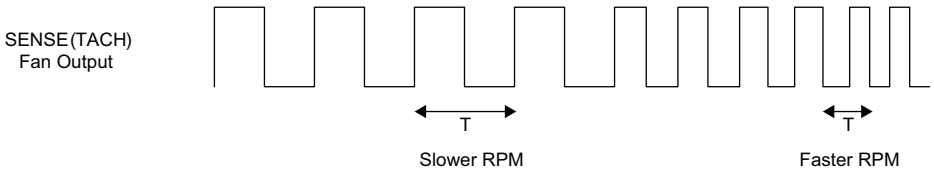
- **Alarm (Active Low)** – Fans with an active high sense output follow the behavior shown in Figure 14. If the fan detects an underspeed or lock condition, the alarm output is driven high (The definition of an underspeed condition can be found in the fan data sheet). The fan drives the output low when the underspeed or lock condition is no longer detected. The fan controller component will immediately assert the ALARM OUT signal to the user logic when the sense input at the FPGA is driven high.

Figure 14. Alarm (Active Low) Sense Type



- **Tach (2 pulses/rev)** – Fans with a tachometer sense output provide sense feedback about the rotational speed of the fan. The tachometer output is a square wave signal, with the frequency of the square wave varying proportionally to the rotations per minute (RPM) of the fan. Depending on the construction of the fan, the tachometer output pulse may present two square wave pulses per rotation, four pulses, or more. The fan controller component is designed to work with 2 pulses/revolution fans across a wide range of speeds. Tachometer outputs with more pulses/revolution will also work with the component, dependent on the fan RPM. Figure 15 shows a typical tachometer output, displaying the relationship between the Tachometer signal and the RPM.

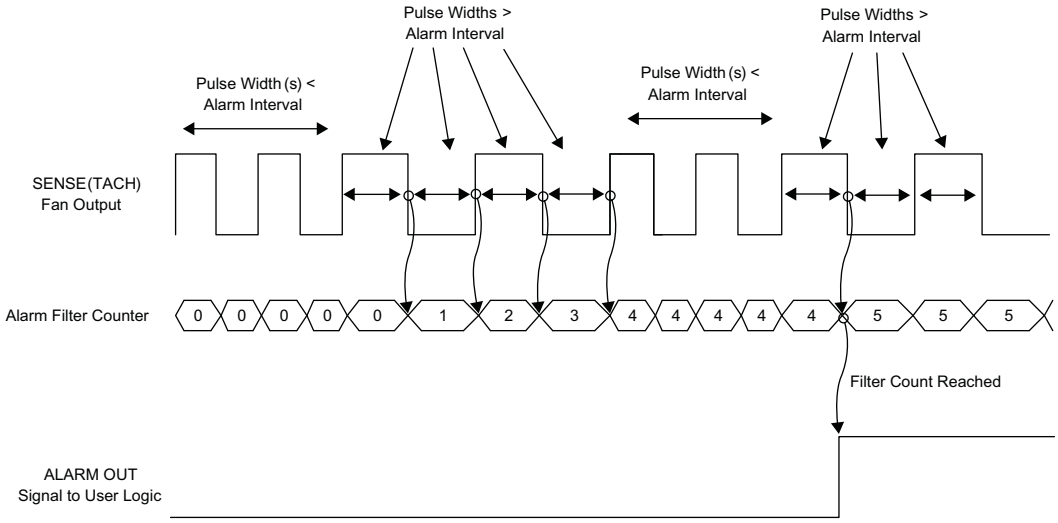
Figure 15. Tachometer Output Signal



Alarm Interval (ms) – The alarm interval setting only applies to the Tach (2 pulses/rev) Sense Type. The alarm interval is used by the fan controller component to alarm the user logic of an underspeed or stuck fan condition. The alarm interval time setting is the pulse width of the tachometer output above which an underspeed or stuck fan condition will be reported to the user logic (dependent on the alarm filter setting, see below). The alarm interval and filter count settings are shown in relationship to the ALARM OUT signal in Figure 16. The allowed alarm interval setting range is 1 ms to 100 ms, with a resolution of 1 ms.

Alarm Filter Count – The Alarm Filter Count setting is used together with the Alarm Interval setting to determine the tachometer output conditions which will alarm the user logic of an underspeed or stuck fan condition. The filter counter is a cumulative counter. Each time the underspeed detector measures a pulse with pulse width greater then Alarm Interval, it will increment the Alarm Filter Counter. When the Alarm Filter Counter reaches the Alarm Filter Count setting, the underspeed detector will assert the ALARM OUT signal to the user logic. The allowed setting range for the filter count is from 1 (no filter applied) to 10.

Figure 16. Alarm Generation Based on Alarm Interval and Alarm Filter (Filter Count = 5)



Startup Control – The startup control check box will enable a “kickstart” pulse when the fan is started. The kickstart pulse will be applied whenever the FAN_OFF signal (see interconnections section) transitions from 1 to 0.

Pulse Duration – This setting defines the length of the kickstart pulse when startup control is enabled. The pulse duration setting range is from 0.25s to 2s, in increments of 0.25s.

Speed 3 / Speed 2 / Speed 1 – The speed settings are PWM duty cycles which the fan controller outputs on the PWM OUT dependent on the Speed Sel 1 and Speed Sel 0 signals (see the interconnections section for more details). Speed 3 is always the highest speed, with Speed 2 the middle speed, and Speed 1 the lowest speed. The duty cycles may be set in 5% increments from 5% to 95%.

PWM Frequency – The PWM frequency may be set between 50 Hz and 80 kHz. There are ~70 discrete frequencies which may be selected.

The PWM frequency is determined by the fan used in the application and the control circuit. 4-wire fans are specified to work with a range of duty cycles shown in their data sheets. These frequencies are typically 20 kHz and higher. 2-wire and 3-wire fans do not specify the frequencies, as this is applied externally. The frequency for switching these fans is often lower (<1kHz). Applying a frequency which is too high may result in a fan which does not switch properly, due to the internal fan circuitry.

PWM Polarity – This setting determines the relationship between the speed setting and the PWM output. Active high means that the assigned speed setting is the high time duty cycle. Active low means that the assigned speed setting is the low time duty cycle. See Figure 17 for an example of the active high and active low duty cycles.

Figure 17. PWM Polarity Setting

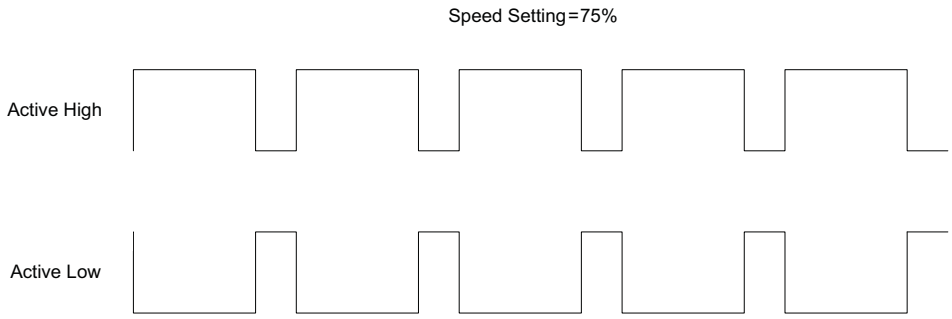
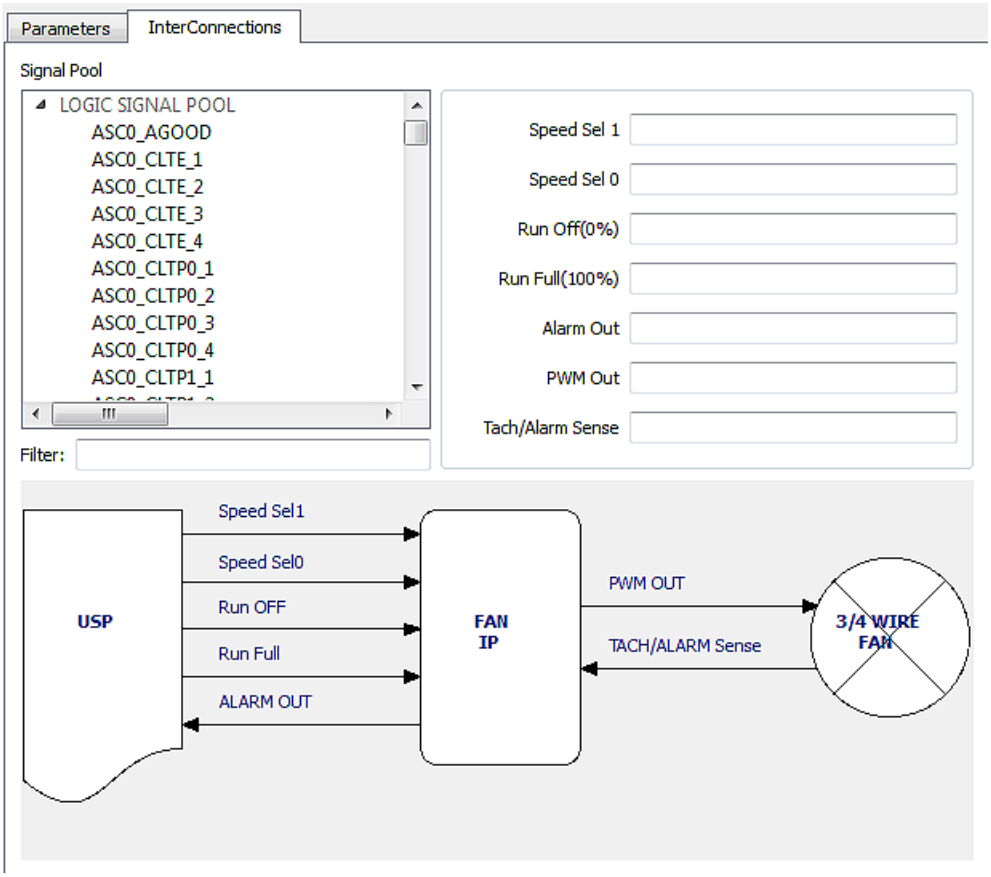


Figure 18. Fan Controller Component - Interconnections



The interconnections tab is used to connect the fan controller IP component to the user logic and the I/O ports connected to the fan circuit. The interconnections are detailed below:

User Logic Nodes

- **Speed Sel 1 / Speed Sel 0** – These nodes are used to select the different speed settings on the PWM output to the fan. The selected speeds based on the settings are shown in Table 2.
- **Run Off (0%)** – This node is used to turn the PWM output off (a 0% duty cycle, as defined by the PWM polarity setting). The behavior of the PWM output compared to this signal is shown in Table 2.
- **Run Full (100%)** – This node is used to turn the PWM output full on (a 100% duty cycle, as defined by the PWM polarity setting). The behavior of the PWM output compared to this signal is shown in Table 2.

Table 2. PWM Output vs Control Signal Settings

Run Off	Run Full	Speed Sel 1	Speed Sel 0	PWM Output
1*	X	X	X	0%
0*	1	X	X	100%
0	0	0	0	Speed 1
0	0	0	1	Speed 2
0	0	1	X	Speed 3

*- If Startup Control is enabled, changing Run Off = 1 to Run Off = 0 will result in PWM Output of 100% for the defined Pulse Duration

- **Alarm Out** – This node is provided from the fan controller component to the user logic. The fan controller component uses this node to signal an underspeed or stuck fan condition. The user logic can react to the fan status accordingly. See the parameter section for more details.

I/O Ports

- **PWM Out** – Output port from the FPGA to the fan controller circuit, used to control the speed in 2, 3, and 4 wire fans.
- **Tach/Alarm Sense** – Input port to the FPGA from the fan. Used to provide speed or locked rotor information in 3 and 4 wire fans.

Connecting the Temperature Monitor and Fan Controller – Integrating Thermal Management

This section will provide a step-by-step flow for implementing a simple thermal management algorithm. This example can be demonstrated on the Platform 2 EVB if available (This document does not cover creating and building your project, or programming the EVB – see the related documents section). There are 4 major steps to implementing the thermal management algorithm in Platform Designer:

1. Temperature Monitor Setup  Temperature
2. Implementing the Ports and Nodes to Connect to the Fan Controller  Ports & Nodes
3. Fan Controller Setup  Fan Controller
4. Implementing the Logic Design to Manage the Fan Controller  Logic

Temperature Monitor Setup  Temperature

The temperature monitor parameters are detailed in the Temperature Monitor Features section. The trip point thresholds and sensor configuration must be updated at minimum to use the temperature monitors in an application. The temperature spreadsheet interface is used to enter the settings for a given application. Settings are made in the editable cells of the spreadsheet by text entry or choosing from a drop down menu. This example works with the following settings (Table 3 and Table 4) for ASC0_TMON2:

Table 3. Temperature Monitor Setup - Part 1

ASC Device	Pin	Schematic Net Name	Logical Name	Monitoring Type	Trip Point Selection	Hysteresis	Monitor Alarm Filter
ASC0	TMON2	TEMP_SENSE2	TEMP_SENSE2_OT_FAULT	OT	30	5	8
			TEMP_SENSE2_OT_WARN	OT	40	5	4

Table 4. Temperature Monitor Setup - Part 2

ASC Device	Pin	Measurement Averaging Filter Coefficient	Offset	Short Fault Measurement Reading	Ideality Factor	Sensor Configuration
ASC0	TMON2	1	0	255.75	1.0079	Beta Compensated PNP

Ports and Nodes Implementation Ports & Nodes

The ports and nodes window is used to create node signals for connecting user logic and components, assign labels and functions to the FPGA PIO, and configure the GPIO and HVOUT ports of the ASC and Platform Manager 2.

Nodes should be added to the design to connect the logic and components. The Nodes tab is a spreadsheet view which by default lists a set of nodes which are provided for interfacing to functions in the ASC such as trimming or fault logging. Additional nodes can be inserted by right clicking in the spreadsheet, or by highlighting a row and pressing the insert key. The logical name can be updated to simplify the design process.

For this example, the following nodes (shown in Figure 19) should be inserted:

Figure 19. Inserted Nodes for Thermal Management Algorithm

	Logical Name	Reserved Function	Group By	Register Type	Reset Level
1				Registered	Don't Care
2	Fan2_SpeedSel1	N/A	N/A	Registered	Set Low
3	Fan2_SpeedSel0	N/A	N/A	Registered	Set Low
4	Fan2_RunOff	N/A	N/A	Registered	Set High
5	Fan2_RunFull	N/A	N/A	Registered	Set Low
6	Fan2_AlarmOut	N/A	N/A	Registered	Set Low

The Ports tab is a spreadsheet view which includes rows for each available PIO in the FPGA. Logical names and I/O settings can be assigned in the ports tab. For the example (shown in Figure 20), the ports should be updated as below:

Figure 20. Updated Ports for Thermal Management Algorithm

	PIO Count	Logical Name	Group By	Type	Register Type	Reset Level
1				INOUT	Registered	Don't Care
2	1	ASC0_RSTN	N/A	IN	Registered	Don't Care
3	2	ASC0_CLK	N/A	IN	Registered	Don't Care
4	3	Global_Clock	N/A	GCLOCK	Registered	Don't Care
5	4	Fan2_PWMOUT	N/A	OUT	Registered	Set Low
6	5	Fan2_TachSense	N/A	IN	Registered	Don't Care
7	6	Manual_Restart	N/A	IN	Registered	Don't Care

If you plan to run the temperature monitoring and fan control example on the demo board, you should also configure GPIO4 and HVOUT4 as shown in Figure 21 and Figure 22. On the Platform Manager 2 EVB these outputs can be used to heat up a resistor which is mounted on top of temperature sensor 2.

Figure 21. Updated GPIO for Thermal Management Algorithm

	ASC Device	Pin	Logical Name	Type	Reset Level
1	ALL	GPIOs		OUT	Set Low
2	ASC0	GPIO1	ASC0_GPIO1	OUT	Set Low
3	ASC0	GPIO2	ASC0_GPIO2	OUT	Set Low
4	ASC0	GPIO3	ASC0_GPIO3	OUT	Set Low
5	ASC0	GPIO4	HEATER_SUPPLY_ON	OUT	Set Low
6	ASC0	GPIO5	ASC0_GPIO5	OUT	Set Low
7	ASC0	GPIO6	ASC0_GPIO6	OUT	Set Low
8	ASC0	GPIO7	ASC0_GPIO7	OUT	Set High
9	ASC0	GPIO8	ASC0_GPIO8	OUT	Set High
10	ASC0	GPIO9	ASC0_GPIO9	OUT	Set High
11	ASC0	GPIO10	ASC0_GPIO10	OUT	Set Low

Figure 22. Updated HVOUT for Thermal Management Algorithm

	ASC Device	Pin	Logical Name	Output Setting	Voltage (V)	Source Current (uA)	Sink Current (uA)	Output Mode	Frequency Select (kHz)	Duty Cycle	Reset Level
1	ALL	HVOUTs		Charge Pump	6	12.5	100	Static	31.25	81.25%	Set Low
2	ASC0	HVOUT1	ASC0_HVOUT1	Charge Pump	6	12.5	100	Static	31.25	81.25%	Set Low
3	ASC0	HVOUT2	ASC0_HVOUT2	Charge Pump	6	12.5	100	Static	31.25	81.25%	Set Low
4	ASC0	HVOUT3	ASC0_HVOUT3	Charge Pump	6	12.5	100	Static	31.25	81.25%	Set Low
5	ASC0	HVOUT4	HEATER_ON	Charge Pump	10	100	100	Static	31.25	81.25%	Set Low

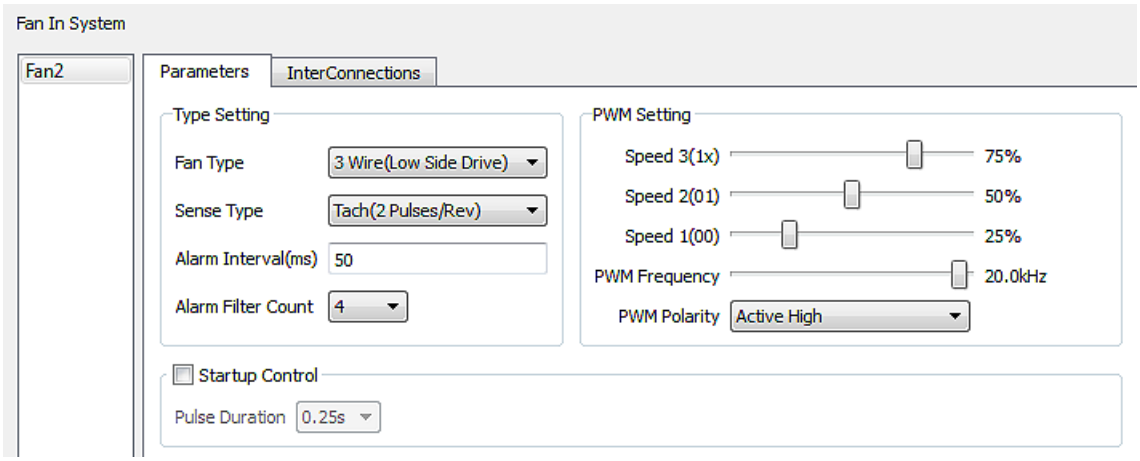
Fan Controller Setup

Fan Controller

The fan controller parameters and interconnections are described in the Fan Controller Features section. For the example design, start by clicking the add fan button in the lower left of the Fan Controller component view.

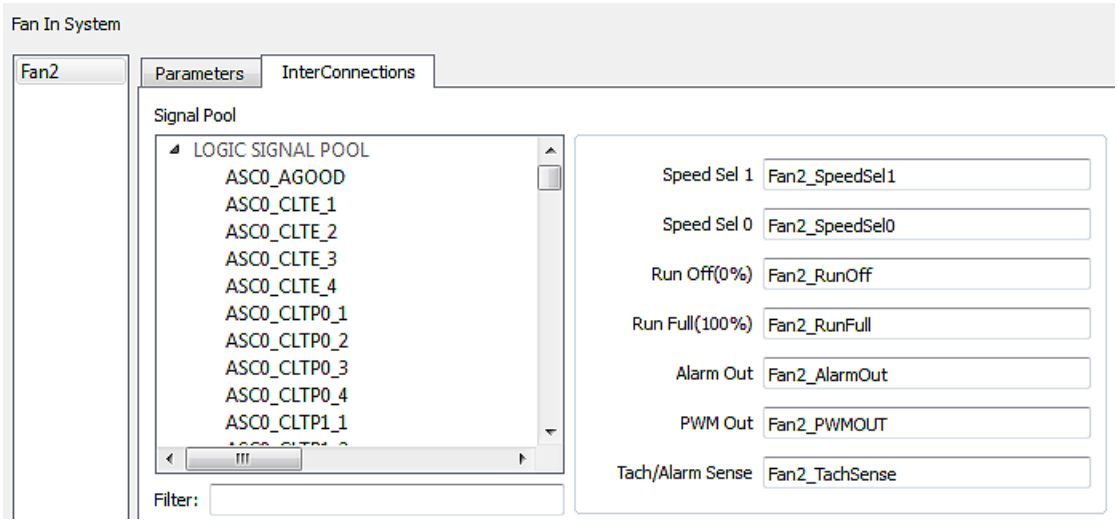
In the example, we rename the fan “Fan2” as this corresponds to the fan connector on the Platform Manager 2 EVB which is used in the demonstration. Set the parameter settings to match those shown in Figure 23 (these correspond to the settings which function best with the fan included with the Platform Manager 2 EVB).

Figure 23. Fan Controller Parameter Settings



The next step is to map the fan interconnections to the ports and nodes setup previously. In the interconnections tab, you can drag and drop signals from the logic signal pool to the empty signal boxes for each interconnection. If you enter “Fan” or “Fan2” into the filter box, the logic signal pool will show only the ports and nodes you previously setup. Set the interconnections to those shown in Figure 24.

Figure 24. Fan Interconnection Settings



Now the fan controller component is configured and ready to be integrated with the logic design.

Logic Design Implementation Logic

The logic design view provides an interface for implementing state machine sequences, logical equations, creating timers, and importing user HDL to a design. In this example, a very simple sequence is used to demonstrate the thermal management integration into the hardware management control. Details on working with the logic sequencer can be found in the References section.

Before implementing the thermal management sequence, a timer must be created. From the logic view, click the “Timers” tab at the bottom of the view. Click the add button to add a new timer. You can rename the timer to “Fan Timer”, and change the period to 500 ms. For the example you can change the clock source to EFB Prescaler. The full timer setting is shown in Figure 25

Figure 25. Logic - Timer Setting

#	Timer Name	Clock Source	Period	Resource(LUTs)
1	FanTimer	EFB Prescaler	500ms	6

Now you can go back to the sequence tab and build a simple thermal management sequence. You can add steps to the sequence by double-clicking the “Begin Shutdown Sequence” step or by pressing the insert key. After inserting the new step, you can double click that step to edit it. The help button in the edit dialog box for each step will guide you through each step. Build the sequence shown in Figure 26.

Figure 26. Logic - Thermal Management Sequence

Step	Sequence Instruction	Outputs	Interruptible	Comment
SM0 Step 0	Begin Startup Sequence		no	
SM0 Step 1	Wait for NOT Manual_Restart		no	Demo starts with push button press on the Platform Manager 2 EVB
SM0 Step 2	HEATER_ON = 1, HEATER_SUPPLY_ON = 1		yes	Turn the current source to the heater resistor on in order to heat up the Temperature Sensor
SM0 Step 3	Wait for TEMP_SENSE2_OT_WARN	Fan2_RunFull = 0, Fan2_RunOff = 0, Fan2_SpeedSel0 = 0, Fan2_SpeedSel1 = 0	yes	Turn the fan on, to speed 1. Wait for a temperature warning.
SM0 Step 4	Wait for TEMP_SENSE2_OT_FAULT	Fan2_RunFull = 0, Fan2_RunOff = 0, Fan2_SpeedSel0 = 1, Fan2_SpeedSel1 = 0	yes	Temperature warning detected, wait for a temperature fault.
SM0 Step 5	Wait for NOT (TEMP_SENSE2_OT_FAULT OR TEMP_SENSE2_OT_WARN)	Fan2_RunFull = 0, Fan2_RunOff = 0, Fan2_SpeedSel0 = 0, Fan2_SpeedSel1 = 1	yes	Temperature fault detected. Turn to speed 3 and wait for fault to clear.
SM0 Step 6	Go to step 3		yes	Temperature fault and warning have cleared, return to step 1.
SM0 Step 7	Fan2_RunFull = 1, Fan2_RunOff = 1		no	Turn the fan on full speed, to try and dislodge any obstruction in the fan.
SM0 Step 8	Wait for Fan2_AlarmOut or wait for 500ms using FanTimer If Timeout Then Goto 1 with { Fan2_RunOff = 1, HEATER_ON = 0, HEATER_SUPPLY_ON = 0 }		no	If fan alarm signal is not cleared, fan is still locked. Shut off the fan and heater and return to the start of the thermal management sequence.
SM0 Step 9	Go to step 3	Fan2_RunFull = 0	no	Fan alarm signal has been cleared by the controller - the underspeed condition is no longer detected. Return to the normal operating routine.
SM0 Step 10	Begin Shutdown Sequence		no	
SM0 Step 11	Halt (end-of-program)		no	

Exception ID	Boolean Expression	Outputs	Exception Handler	Comment
E 0	IF Fan2_AlarmOut		Start at step 7	If an underspeed condition is detected, go to underspeed handler.
	<end-of-exception-table>			

SM0 +

This completes the thermal management algorithm implementation in platform designer. See the References section for information on building a complete design and the Platform Manager 2 EVB.

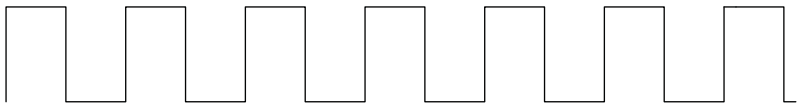
3-Wire Fan – Considerations for Underspeed Detection

Controlling the speed of 3-wire fans with a PWM signal interrupts the power supply to the fan. During this interruption, the feedback signal (Tachometer or Locked Rotor Alarm) will also be interrupted. Two potential ways to maintain an underspeed detection mechanism in this scenario are pulse stretching or using a hold-up capacitor to maintain the feedback circuit supply. The fan controller component will automatically implement pulse stretching if a 3 wire (Low Side Drive) fan type is selected. If a 3 wire (High Side Drive) fan type is selected, the fan controller underspeed detection will only function with a properly sized hold up capacitor. The low side drive implementation is simpler in terms of circuit board components, but the pulse stretching function will consume additional LUTs in the hardware management controller FPGA. The high side drive circuit requires additional circuit board components but uses less LUTs. Pulse stretching and sizing of the hold up capacitor are described in the following sections.

Pulse Stretching

The fan controller component uses the feedback signal (tachometer or locked rotor alarm) to determine if an underspeed condition has occurred. The tachometer output for a fan with an uninterrupted power supply is shown in Figure 27.

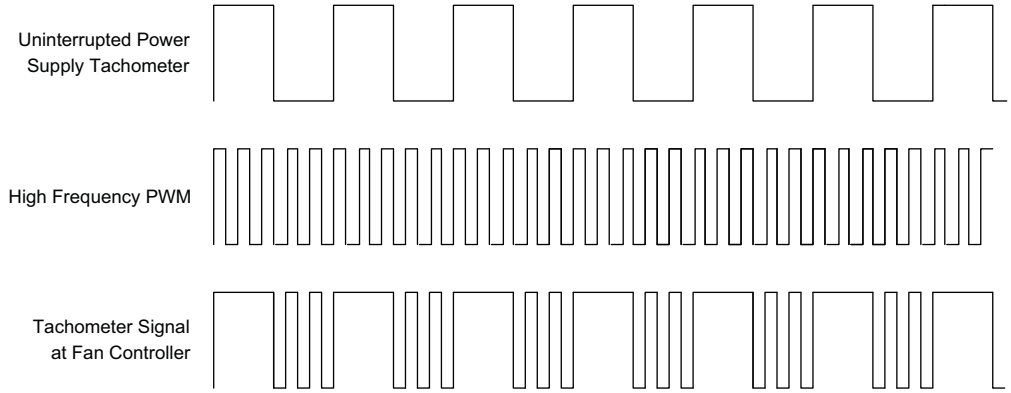
Figure 27. Tachometer Output with Uninterrupted Power Supply



PWM control of the fan will interrupt the power supply to both the fan and the feedback circuit. This will result in an interruption of the feedback signal. The tachometer output presented to the fan controller will be the ideal tachometer signal (as shown in Figure 27) overlaid with the PWM control signal. Figure 28 shows the behavior with a high

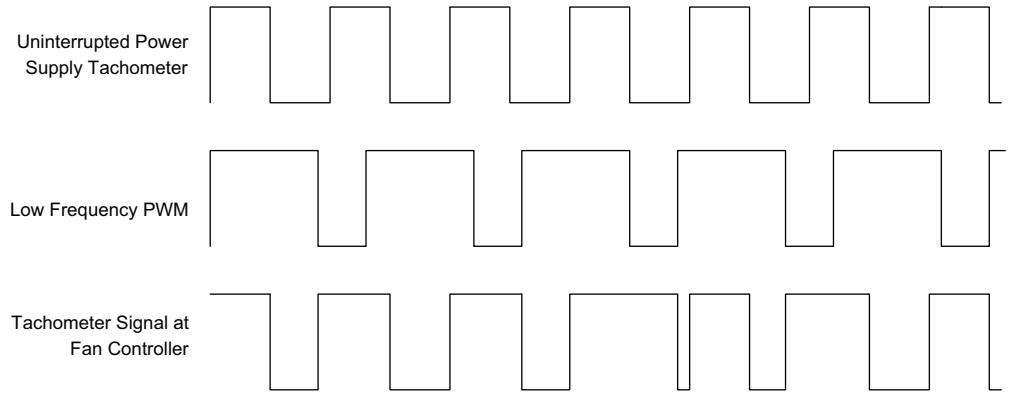
frequency PWM control signal.

Figure 28. Tachometer Output with High Frequency PWM Overlaid



When a low frequency PWM control signal is used, the tachometer signal will present slightly differently. The signals are still overlaid, however the pattern appears more random, as shown in Figure 29.

Figure 29. Tachometer Output with Low Frequency PWM Overlaid



The overlaid tachometer signals make it impossible for the fan controller to reliably detect underspeed conditions. The tachometer signal transitions which are caused by the PWM signal will be falsely captured by the fan controller and corrupt the underspeed detection hardware.

The fan controller will apply pulse stretching in order to recover the “real” tachometer signal. The controller will periodically apply a 100% duty cycle PWM (fixed on) and measure the resulting uninterrupted tachometer signal. Figure 30 shows the applied PWM and resulting tachometer signal at the fan controller for a high frequency PWM. Figure 31 shows the applied PWM and resulting tachometer signal at the fan controller for a low frequency PWM.

Figure 30. Tachometer Signal with Pulse Stretching - High Frequency PWM

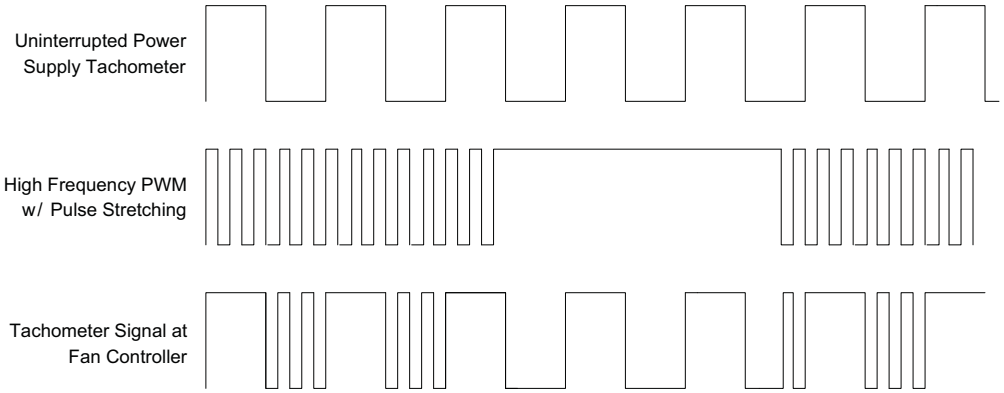
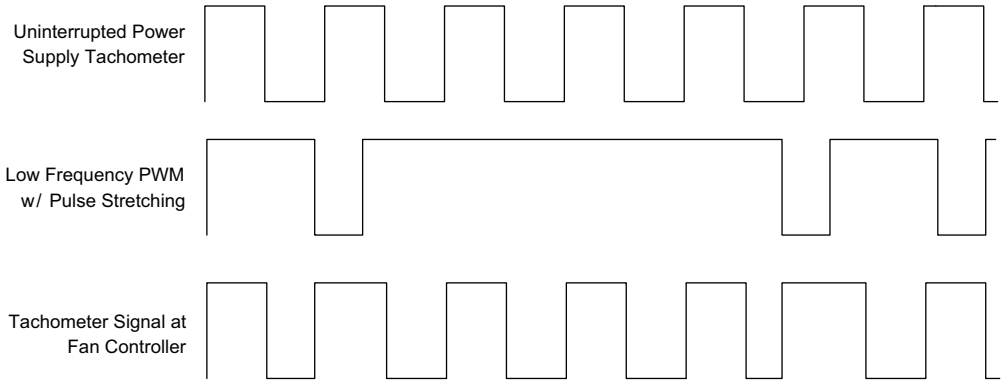


Figure 31. Tachometer Signal with Pulse Stretching - Low Frequency PWM



The pulse stretching is applied only periodically, in order to maintain the applied PWM speed control for the majority of the fan operation. The delay between pulse stretching measurements and the length of the pulse are dependent on the chosen ALARM INTERVAL (see Fan Controller Features section). This dependency guarantees that the pulse stretching will not exceed 4% of the overall operation time, which limits the negative effects on the speed control. The delay is shown in Table 5. The pulse stretching duration is 2 times as long as the chosen alarm interval, in order to guarantee that an underspeed condition will be detected.

Table 5. Pulse Stretching Parameters

Alarm Interval Range (ms)		Pulse Stretching Duration (ms) – (2 x Alarm Interval)		Delay between pulse stretches (seconds)
Minimum	Maximum	Minimum	Maximum	
1	10	2	20	1
11	30	22	60	2
31	50	62	100	3
51	70	102	140	4
71	100	142	200	5

Hold-Up Capacitor Sizing

The 3-wire (High-Side Drive) fan type requires a hold-up capacitor to guarantee proper underspeed detection. The capacitor is used to hold-up the feedback sensor supply during the PWM off time to the fan. The capacitance value selected is determined by multiple factors, with the primary ones being the selected PWM frequency and the supply current required by the fan.

If the chosen capacitance is too small, the sensor supply will drop out and the overlay effect of the PWM control signal will be present, as shown in the previous section. A smaller capacitance may also limit the minimum duty cycle which can be used with the fan control. If the capacitance is too large, the speed control using the PWM signal will not be effective.

In general, a higher PWM frequency is desirable as this reduces the size of the hold-up capacitor. In the Platform Manager 2 EVB, a set of capacitors, connected through jumpers, are provided to allow capacitance size matching to the PWM frequency. These are optimized to the fan provided with the EVB. This fan is a 5V DC fan from Delta Electronics, rated for 9000 RPM at 5V with a current draw of 160 mA (part #: AFB0305HA-AF00). Table 6 shows the capacitance values versus frequency ranges recommended for this fan.

Table 6. Hold-up Capacitance vs PWM Frequency (Platform Manager 2 EVB)

PWM Frequency Range		Hold-Up Capacitor Size
Minimum	Maximum	
40 kHz	80 kHz	0.22 uF
10 KHz	26.7 kHz	2 uF
800 Hz	10 kHz	10 uF

When selecting a capacitance value for use with your own fan, you can use these values as a starting point. Fans which draw more current at similar voltage levels will require larger capacitance values.

Summary

The combination of programmable temperature monitors and fan controller components make Platform Manager 2 designs ideal solutions for thermal management. The programmable parameters for both the temperature monitor and fan components have been described, along with the steps to integrate these components into your hardware management design.

The information covered in this technical note can be applied to all Platform Manager 2 designs. Platform Manager 2 designs can be built using several combinations of Lattice devices as listed in Table 7.

Table 7. Lattice Platform Manager 2 Design Options

Central Hardware Manager	Hardware Management Expander	Number of Expanders Supported[1, 2]
LPTM21	LPTM21L or L-ASC10	0 – 3
LPTM21L	LPTM21L or L-ASC10	0 – 3
MachXO2	LPTM21L or L-ASC10	1 – 8
MachXO3	LPTM21L or L-ASC10	1 – 8
ECP5	LPTM21L or L-ASC10	1 – 8

1. Platform Manager 2 designs with 6 hardware expanders are best supported with MachXO2 and MachXO3 devices of 2k LUTs or larger.
2. Platform Manager 2 designs with 8 hardware expanders are best supported with MachXO2 and MachXO3 devices of 4k LUTs or larger.

References

- FPGA-DS-02038 (previously DS1042), L-ASC10 Data Sheet
- FPGA-DS-02036 (previously DS1043), Platform Manager 2 Family Data Sheet
- DS1035, MachXO2 Family Data Sheet
- FPGA-DS-02032 (previously DS1047), MachXO3 Family Data Sheet
- FPGA-DS-02012 (previously DS1044), ECP5 and ECP5-5G Family Data Sheet
- EB93, Platform Manager 2 Evaluation Board User Guide
- Platform Designer 3.1 User Guide

Technical Support Assistance

For assistance, submit a technical support case at www.latticesemi.com/techsupport.

Revision History

Date	Version	Change Summary
September 2018	1.2	Changed document number from TN1278 to FPGA-TN-02080.
		Included references to LPTM21L, MachXO3, and ECP5.
		Updated Introduction section. Added Table 1 and Figure 1.
		Updated Summary section. Added Table 7.
		Updated References section. Added documents and updated document numbers.
		Updated Technical Support Assistance.
August 2014	1.1	Corrected Figure 6, Monitor Alarm Signal Hysteresis - Undertemperature (UT) Setting.
		Updated PWM Frequency description.
December 2013	01.0	Initial release.