Window Lift and Relay Based DC Motor Control Reference Design Using the S12VR Microcontrollers

Document Number: DRM160
Rev. 1
04/2015

freescale™

Contents

Section number **Title** **Page**

Chapter 1
Overview

This reference manual describes the design of a relay-driven DC motor controller using a Hall sensor for position feedback. The use case of this reference design is made to address automotive window lifts, but it can alternatively be used as a relay DC motor controller.

The design exhibits the suitability and advantages of the MagniV S12VR microcontroller for relay driven motor control. It serves as an example of an anti-pinch application using the MagniV S12VR series of microcontrollers.

The MC9S12VR is an optimized automotive 16-bit microcontroller focused on low cost, high performance, and low pin count. This device integrates an S12 microcontroller with a LIN physical interface, a 5 V regulator system, and analog blocks to control other elements of the system which operate at vehicle battery level (relay drivers, high side driver outputs, wakeup inputs). The MC9S12VR is targeted at generic automotive applications requiring single node LIN communications. Typical examples of these applications include window lift modules, seat modules, sunroof modules, etc.

The following features make the MC9S12VR ideal for reducing system cost, bill of materials cost, development cost, and saving circuit board space:

- 1 MHz internal RC oscillator with +/-1.3% accuracy over rated temperature range
- One on-chip LIN physical layer transceiver fully compliant with the LIN 2.2 standard & SAE J2602-2 LIN standard
- On-chip voltage regulator for all internal voltages
- Two protected low side outputs to drive inductive loads
- Up to two protected high side outputs
- Four high voltage inputs with wakeup and readable internally on ADC
- Up to two 10 mA high current outputs
- 20 mA high current output for Hall sensor supply
- Battery voltage sense with reverse battery protection

Other useful features of the MC9S12VR are:

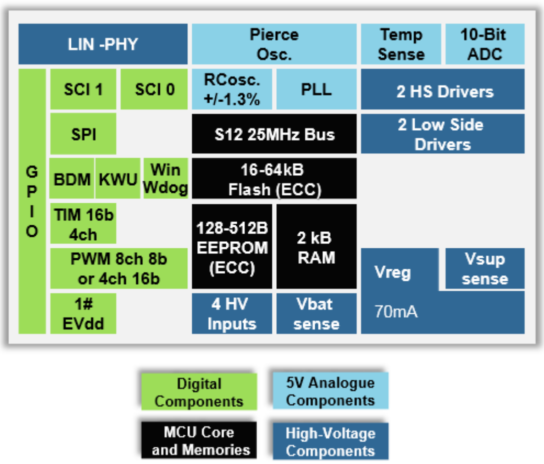
- HCS12 CPU core
- 64 or 48 KB on-chip flash with ECC
- 512 byte EEPROM with ECC
- 2 KB on-chip SRAM
- Phase locked loop (IPLL) frequency multiplier with internal filter
- 4-20 MHz amplitude controlled pierce oscillator
- Internal COP (watchdog) module (with separate clock source)

- Timer module (TIM) supporting input/output channels that provide a range of 16-bit input capture, output compare, and counter (up to four channels)
- 10-bit resolution ADC with up to six channels available on external pins
- One SPI module
- One SCI supporting LIN
- One additional SCI
- Autonomous periodic interrupt
- Chip temperature sensor

These features substantially reduce the cost of developing and manufacturing applications such as:

- Automotive power window lift/sunroof with anti-pinch
- Any relay driven DC motors
- LIN slaves with space constraints

The following image depicts a block diagram of the MC9S12VR, the main component of the window lift reference design:



The following features are implemented in this reference design:

- Motor type: DC, 7A typ/16A max.
- Feedback: Hall encoder, voltage at motor terminals
- Communication protocol: LIN
- Hardware ports:
  - VBAD, GND, LIN
  - Motor up, motor down
  - Hall direction, Hall speed, Hall power, Hall ground
- PCB:
  - 2 layer board, assembly on top side only
  - 2.4 x 1.9 inches (including demo switch and push buttons)
  - BOM parts = 40

Chapter 2
Hardware Description

This chapter explains how the hardware of the reference design was implemented. A detailed description of each module connection is described. The following block diagram illustrates all functions used at a high level:

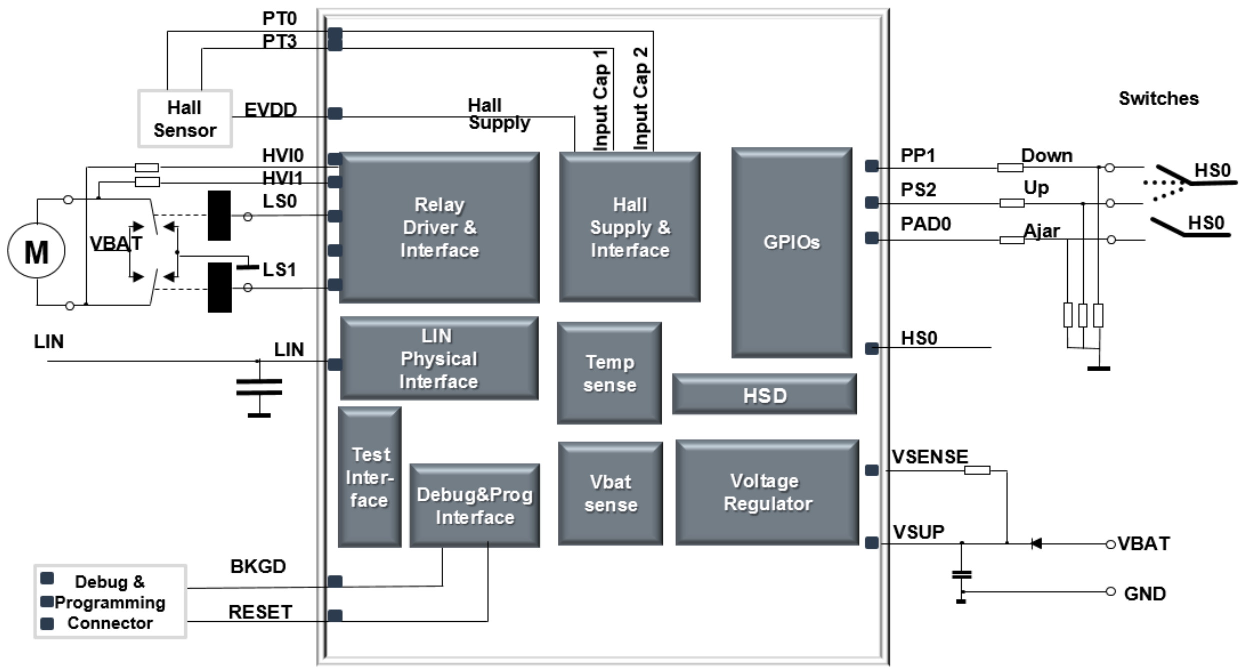


Figure 1. Window lift block diagram

The following paragraphs explain how the different pins and peripherals of the MC9S12VR are connected and used to provide different functions.

Hall sensor: The hall sensor acts as an encoder and outputs square signals that encode speed and direction information. These signals are measured with pins PT0 and PT3 used as input capture channels. The hall sensor is powered by the EVDD pin which is capable of supplying up to 20 mA @ 5 V.

DC motor: The DC motor is driven by a twin single pole double throw relay. This configuration allows the relay to also act as an H bridge allowing the MCU to polarize the motor in either direction. The two coils of the relay are driven by the low side driver 0 and 1 that can be connected directly to the relay coils as they integrate high current capability. They also integrate an internal active clamp that protects the device from coil discharge, without any need for external diodes. The voltage at the motor terminals is monitored by connecting the motor terminals to the high voltage inputs 0 and 1. These HVI inputs allow the ADC to sample this high voltage directly.

Debug: The BDM connector allows the user to reprogram and debug the board. It only requires two pins: RESET and BKGD.

VSUP: A low forward voltage diode protects the rest of the circuit from reverse battery conditions. Two 10 µF/50 V/1210 capacitors are used to decouple transients in the 100 KHz to 1 MHz range.

VSENSE: A 10K resistor and a 6.8 nF/50 V/0402 capacitor make a low pass filter for stable battery measurements.

Motor actuator: The relay takes about 1.3 ms to open or close the terminals (max 5 ms). The relay is activated by two low side drivers. No diode is needed to protect from the relay coil discharge because the low side drivers already include an internal active clamp.

Switches: A three way switch is connected. This switch is for demo purposes as a real device may use a separate panel switch.

The following paragraphs describe how some of the internal modules of the S12VR map to this application and how they are used in the block diagram:

Relay driver and interface: This block uses Low Side Drivers (LSD) 0 and 1 to activate the relay coils. High voltage inputs (HVI) 0 and 1 are used to monitor the motor voltage.

LIN Physical Interface: The S12VR internal LINPHY and the SCI modules are used to implement LIN bus communication without any external components.

Debug & Programming Interface: The BDM module is in charge of implementing debugger tool communication and device reprogramming.

Hall Supply and Interface: Two timer channels in input capture mode implement the Hall encoder interface for measuring motor speed and position. The hall sensor is powered by the high current capability of the EVDD pin.

Voltage Regulator: The S12VR has an integrated automotive voltage regulator that requires only reverse battery protection to connect directly to a 12 V automotive battery.

GPIOs: General purpose pins as inputs to read external switch interface.

The following pages detail the electrical schematics of the window lift reference design.

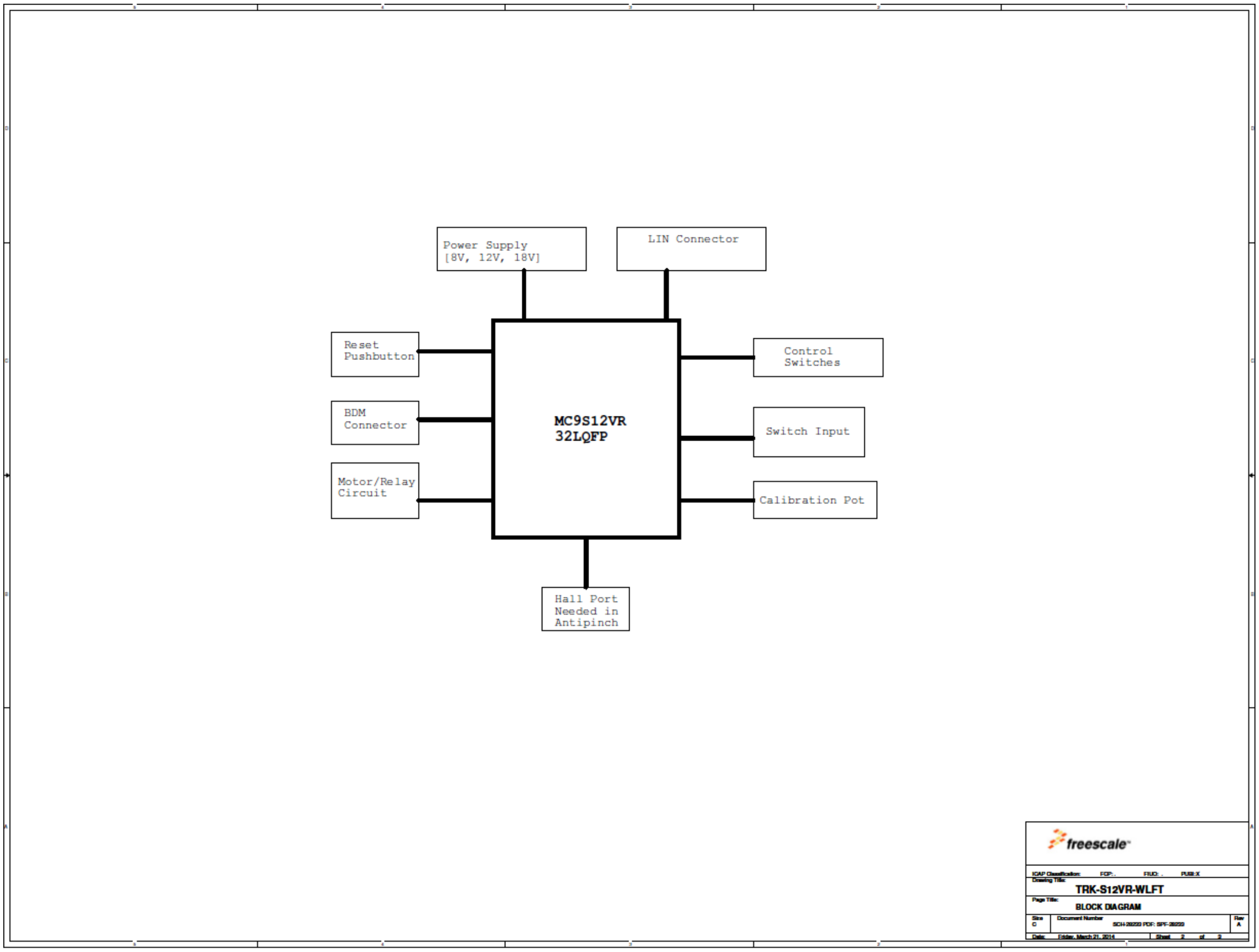


Figure 2. Block diagram

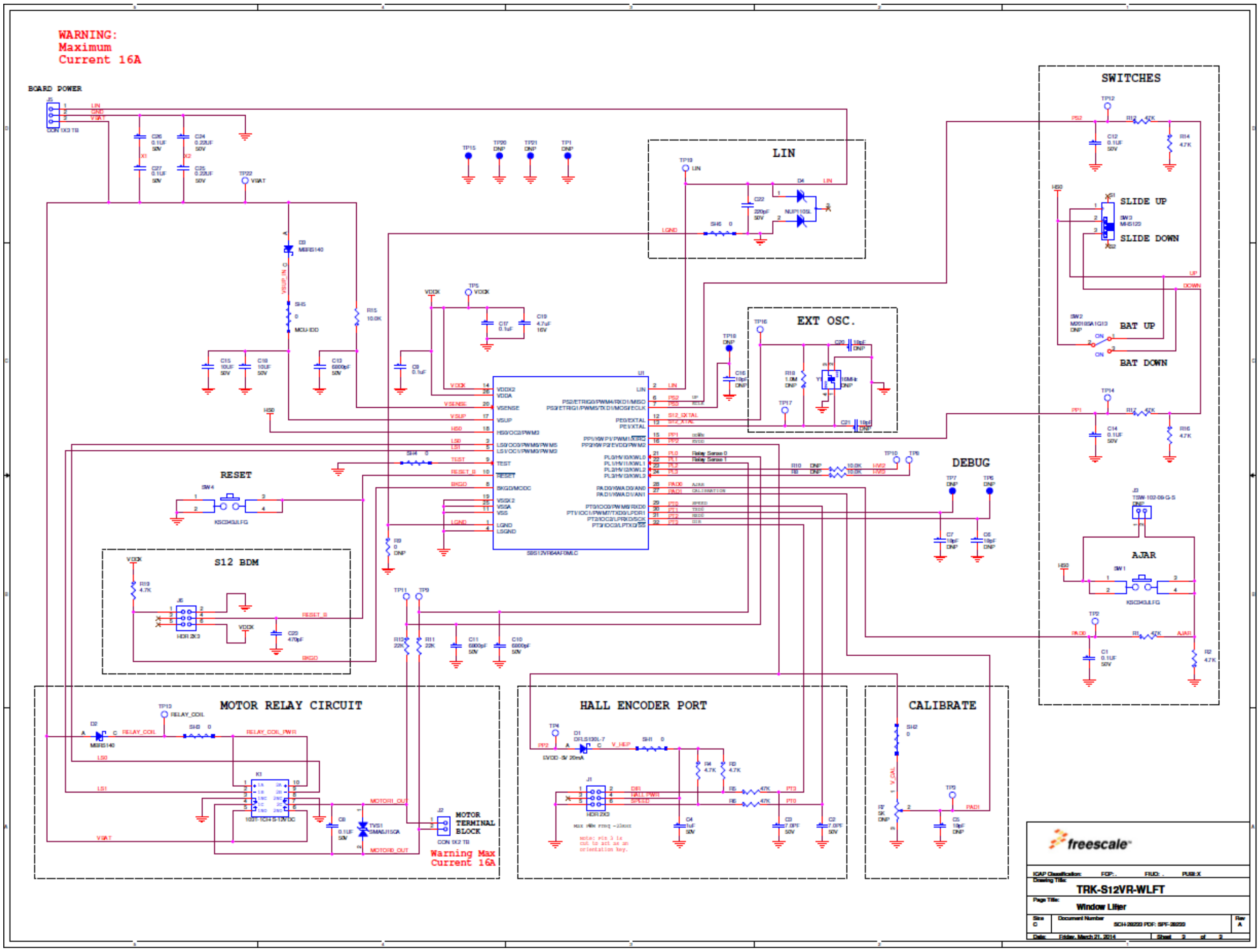


Figure 3. Schematics

Chapter 3
Firmware Description

There are two Codewarrior software projects provided with this reference design (download them by clicking here).

- **Up-down-no-anti-pinch**: This is a very basic software project that allows immediate use of the hardware by implementing a simple up down control with the relay and the input switch. Any DC motor can be connected to the relay terminals to test the board.
- **WindowLift.mcp**: This is a fully functioning window lift controller complete with anti-pinch algorithm.

The features implemented in the WindowLift.mcp Codewarrior project include:

- **Speed and direction:** tracks the window position and the measured period of the speed signal
- **Voltage at the Motor:** Returns the voltage at the motor up/down terminals
- **Expected Speed at 12 V:** Uses the Speed and Direction subsystem to provide an expected speed at 12 V. It depends on a calibration algorithm that stores the window expected speed at each position.
- **Pinch Detection:** Uses the information from the subsystems above to determine the deviation from the computed speed and triggers the self-reversal subsystem when a threshold is passed.
- **Self-Reversal:** When triggered, it controls the self-reversal of the window. It either stops after a certain amount of distance is reversed or after the initial window position was reached.

The firmware that implements the window lift application with the anti-pinch feature is divided into several modules. These modules are illustrated in the following architecture diagram

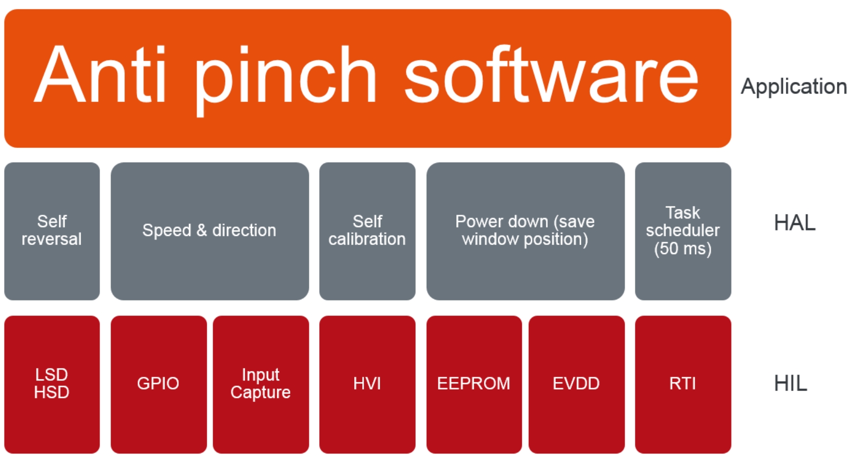


Figure 4. Architecture diagram

There are three main layers HIL, HAL, and application.

The Hardware Interaction Layer (HIL) contains software routines that directly control the S12VR hardware peripherals. These can be seen as low level drivers that provide services to the upper levels in order to implement the complete application. The HIL includes the following drivers:

- **LSD**, **HSD**: Low side drivers and High side drivers for activating the relay.

- **GPIO**: General purpose input output.
- **Input capture**: Configures the timer channels as input capture for measuring the Hall sensor output waveform.

- **HVI**: High voltage inputs.
- **EEPROM**: Electrically Erasable Programmable Read Only Memory.
- **EVDD**: Controls the EVDD pin which provides power to the Hall sensor.
- **RTI**: Real Time Interrupt.

The Hardware Abstraction Layer (HAL) contains software routines that provide higher level services to the application. These routines may use the lower layer drivers to provide a more complex service. The HAL includes the following routines:

- **Self-reversal**: When triggered, it controls the self-reversal of the window. It either stops after a certain amount of distance is reversed or after the initial window position was reached.
- **Speed and direction**: tracks the window position and the measured period of the speed signal.
- **Self-calibration**: The first time the software is executed it drives the window down until it stops. It then drives the window up until it stops at the top, in this way the total number of Hall sensor steps is counted and saved into the EEPROM. With this information the application can distinguish between a motor stalled by reaching the end of the path and a motor stalled by an object in the way.
- **Power down**: Saves the window position before going into a low-power mode.
- **Task scheduler**: executes tasks every 50 ms.

Finally, the application uses all of these routines in conjunction to provide the direction control feature and the anti-pinch feature for a complete window lift application.

The application behavior is described by the following flow diagram:

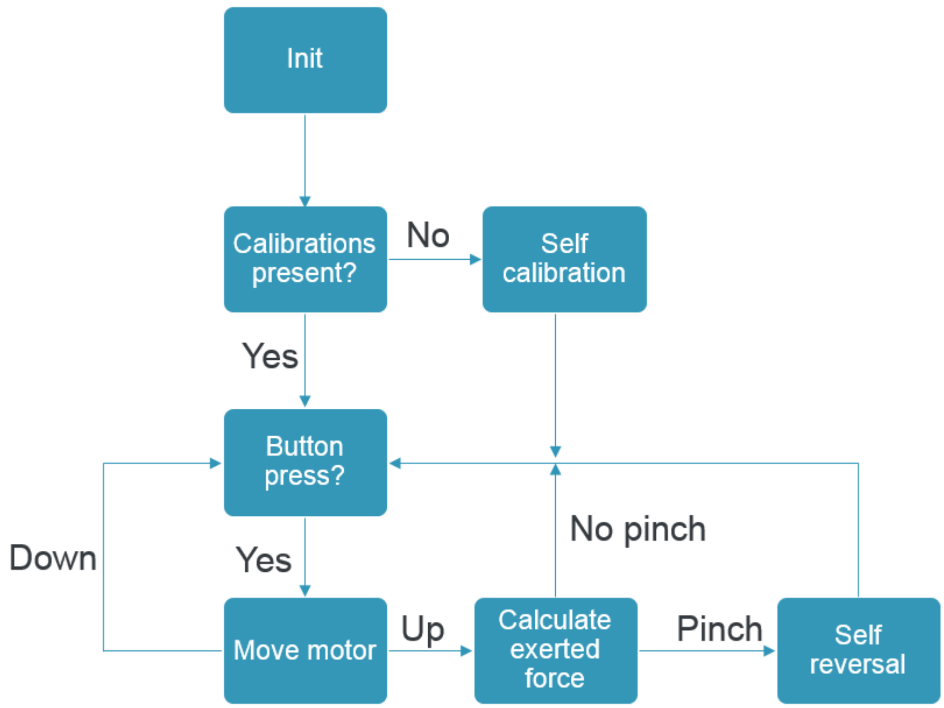


Figure 5. Flow diagram

As it can be seen from the flow diagram, the firmware behavior at a high level is pretty simple. The process can be described as follows:

1. MCU powers on. It initializes all the clocks, peripherals, and GPIO needed.
2. Software checks if calibrations are stored in the EEPROM. If they are not present then self-calibration will be performed.

WARNING

Self-calibration will move the motor down until the window encounters a stall condition (window completely open) and then move the window up until the window encounters a stall again (window completely closed). User must be extremely careful not to obstruct the window during self-calibrate because the anti-pinch safety algorithm is not active during this time.

3. MCU waits until a button is pressed. If the down button is pressed no anti pinch is needed and the window will be moved down until the user stops pressing the down button or if the window reaches the end of the rail. If the up button is pressed the motor moves the window up while the MCU is continuously calculating the exerted force. If the window reaches the end of the rail then it is stopped. If it has not reached the end of the line and an unusually high force is being exerted, then the self-reversal activates, which drives the window down for a couple of seconds in response to a pinch event.

3.1. Software CPU and memory utilization

This reference design provides a good notion of how much resources will be used when implementing a relay driven DC motor application with a software anti-pinch algorithm, such as a window lift, sunroof, or other DC motor application.

- **FLASH**: less than 4 KB. This leaves up to 60 KB free for other uses such as calibrations, complex stacks, drivers, etc.
- **EEPROM**: less than 20 bytes. This leaves up to 492 bytes free to be used by the application for other purposes.
- **RAM**: less than 550 bytes. This leaves 1498 bytes of RAM free for other purposes.

3.2. First time operation

It is recommended that you try the "up-down-no-anti-pinch" software first. This is a very basic software project that allows immediate use of the hardware by implementing a simple up down control with the relay and the input switch. Any DC motor can be connected to the relay terminals to test the board.

WARNING

This software does not have any safety check for stall conditions. If the motor is left running for prolonged times in a stalled condition, permanent damage may occur to the motor.

After the user tests his motor with the "up-down-no-anti-pinch" software, the anti-pinch software can be tested. This requires a motor with a hall sensor and the hall encoder interface to be connected to the board. The software "WindowLift.mcp" has the anti-pinch algorithm implemented.

The first time this software is programmed to the S12VR it will check if calibrations are stored in the EEPROM. The software will perform self-calibration the first time it is programmed.

WARNING

Self-calibration will move the motor down until the window encounters a stall condition (window completely open) and then move the window up until the window encounters a stall again (window completely closed). User must be extremely careful not to obstruct the window during self-calibrate because the anti-pinch safety algorithm is not active during this time.

After self-calibration is performed, the data is stored permanently on EEPROM and the window lift can be used and safely power cycled. The anti-pinch algorithm will detect if any object obstructs the path of the window and reverse the window for a few seconds if a pinch condition was detected.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number DRM160
Rev 1
04/2015