Freescale Semiconductor
Application Note

AN3147
Rev. 0.2, 11/2005

MM908E624 Window Lift / Sun Roof LIN Slave

LIN Connectivity Based on the LIN 1.3 Communication Protocol

by: Petr Cholasta
Roznov Czech System Center, Roznov p.R., Czech Republic

1 Introduction

Many automotive customers are looking to use LIN as the new technology to bring enhancements and more features to the automotive industry. LIN (Local Interconnect Network) is a concept for low cost automotive networks, which complements the existing portfolio of automotive multiplex networks.

The main purpose of this application note is to describe and demonstrate the usage of the MM908E624, which has been developed as a highly integrated and cost-effective solution for driving loads using relays within a LIN architecture. It is especially suited for the control of high-current motors using relays (e.g., window lifts, fans, and sun roofs). The Window Lift application was chosen as one of the typical device utilizations.

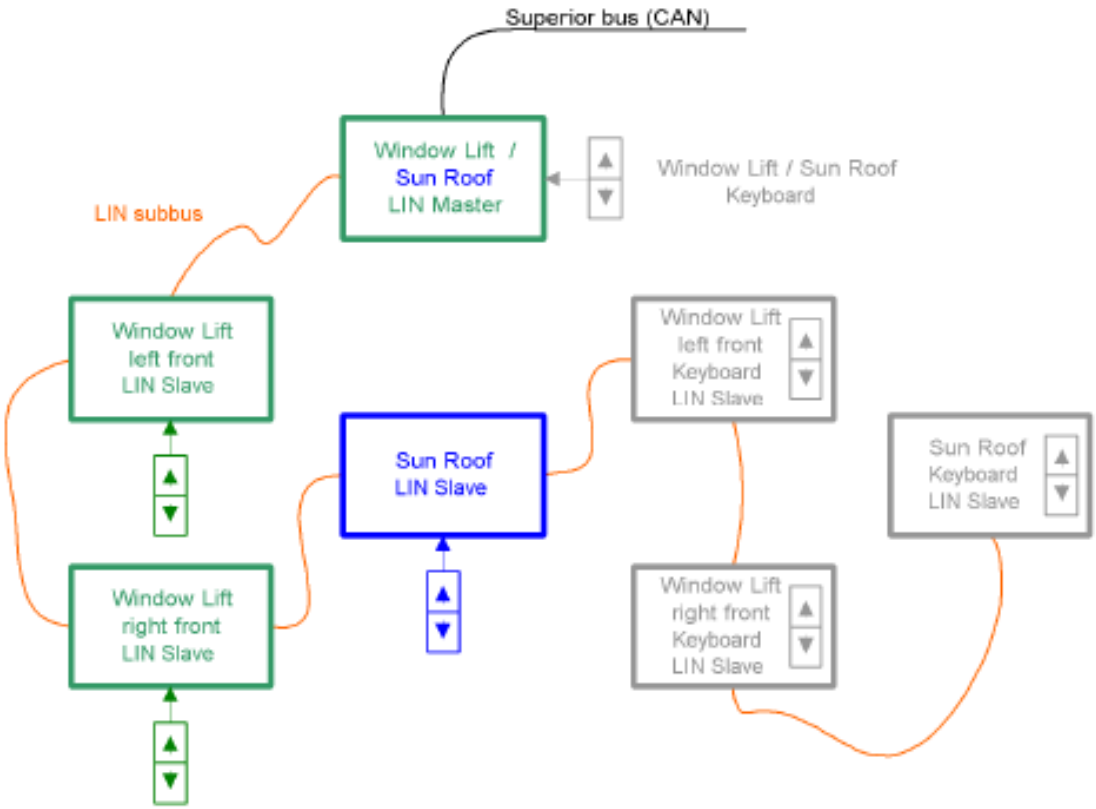
The other goal is to introduce Freescale LIN development tools as the LIN based boards called LINkits and also the FreeMASTER tool as an efficient tool for an application control, evaluation, and visualization.

The theme of this application note is mainly focused on the demonstration of the capabilities and performance of the MM908E624 in a LIN 1.3 enabled design. However, it also introduces the 16-bit MCU MC9S12C32 as a LIN 1.3 Master.

Table of Contents

2 General Description

The Window Lift application outline is based on one of several possible Window Lift / Sun Roof LIN solutions, which Figure 1 displays. The Window Lift / Sun Roof LIN Master controls according to the keyboard Right / Left Window Lift and Sun Roof LIN Slaves. Each Slave can be also controlled by a dedicated LIN Slave keyboard, which can be realized as a standalone LIN Slave or could be part of the Window Lift / Sun Roof LIN Slave node. To support system diagnostics, e.g., the possibility to control and update the whole Window Lift / Sun Roof system, the LIN Master also acts as a gateway to a superior bus, e.g., CAN.



Node in green — covered by the MM908E624 Window Lift application
Node in blue — functionality similar to the MM908E624 Window Lift application (not developed)
Node in gray — other possible Window Lift / Sun Roof Lin nodes and nodes stuff (not developed)

Figure 1. Window Lift / Sun Roof LIN Concept

2.1 System Outline

The Window Lift system concept is displayed in Figure 2. The Window Lift LIN Slave is realized by the MM908E624 single package solution. This device controls the Car Door Window Lift platform by reading the LIN bus data and Hall sensors signal. This enables controlling the window glass position and detecting the window glass antipinch or stall occurrence. The LIN bus data stream is controlled by the LIN Master, realized by the 16-bit MCU (MC9S12C32). This microcontroller was chosen from among the others because of its LIN / CAN gateway capability.

The Window Lift application behavior is controlled by the FreeMASTER tool, which is used for application evaluation also. To allow the user easy application control, GUIs (Graphical User Interface) run in the FreeMASTER tool (see Section 3.1, “FreeMASTER Tool”).

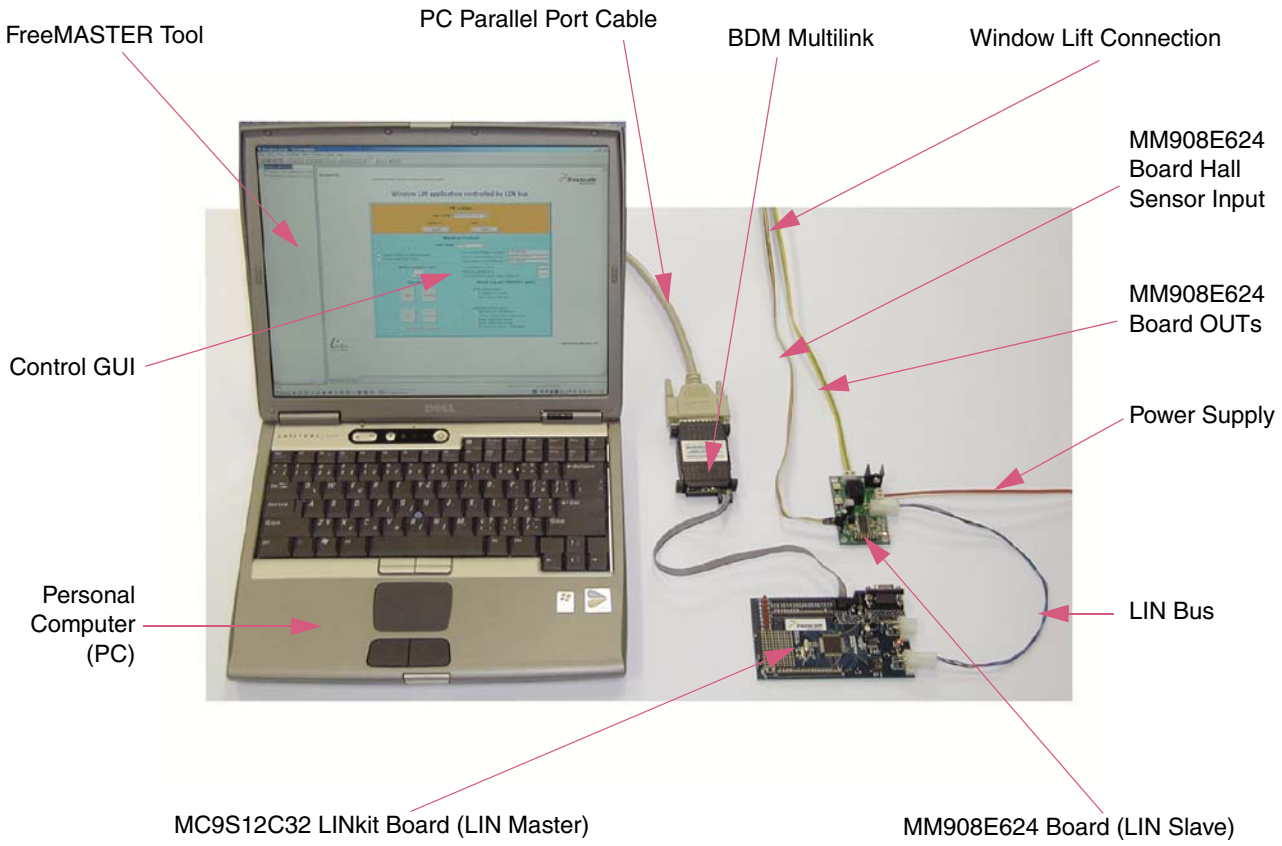


Figure 2. Window Lift System Concept

2.2 System Features

The FreeMASTER tool offers the user application control via three GUIs described in the following sections:

- Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction”
- Section 2.2.2, “GUI “Window Lift Parameters Configuration” Introduction”
- Section 2.2.3, “GUI “The MM908E624 Board OUTs Controlled by the LIN Bus” Introduction”

2.2.1 GUI “Window Lift Application Controlled by the LIN Bus” Introduction

This GUI offers the user to control the MM908E624 board as the Car Door Window Lift application. The system features are as follows:

- LIN bus control:
  - Run / Stop communication
  - Sleep / Wake-up LIN Slaves
- MM908E624 Window Lift LIN Slave control:
  - Select “Normal” or “Keep Window Speed” MM908E624 board mode. During “Keep Window Speed” mode, the window glass movement speed is kept independent of the system power supply voltage variation. The Car Door Window Lift platform has to be adapted to run the “Keep WIndow Speed” mode.
  - Enable / disable window glass soft start / soft stop
  - Enable control of the window glass position by the MM908E624 keyboard, realized by two push buttons, one for “Close Window” and the other for “Open Window” glass movements (see Figure 3).
  - Set the window glass desired position and run the Window Lift mechanism to reach it.
  - “Open / Close” and “Open Completely / Close Completely” window
  - Display current LIN Master control command, current LIN Slave window status, and LIN Slave keyboard request.
  - Display actual window glass position, Hall sensors pulse half period, and system power supply voltage (used for demonstration of window glass movement speed independence of the power supply voltage, when the MM908E624 board is running the “Keep Window Speed” mode).
  - Display window caused stop event (Window antipinch, Hall Port error) and the MM908E624 device analog die status (overvoltage, overtemperature, etc.)

More information on this GUI can be found in the Section 7.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Description”.

2.2.2 GUI “Window Lift Parameters Configuration” Introduction

This GUI is closely linked to that previously described in Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction”. It enables configuration of the Car Door Window Lift application strategic parameters, and in that way it allows the user to easily rebuild the application for another Window Lift platform with similar features as the one used. The GUI offers the following features:

- LIN bus control:
  - Run / Stop communication
  - Sleep / Wake-up LIN Slaves
- Window Lift parameters configuration:
  - Upload Parameters: Load current parameters from LIN Slave to LIN Master
  - Assign Maximal Position: Contents of Window position counter is loaded into the Window maximal position variable. It is used for the Window stop cause (Window antipinch / stall or Hall error)
  - Assign an Antipinch Threshold: Control the antipinch / stall Window mechanism pressure
  - Reset Position Counter: Reset the Window actual position counter
  - Store Parameters To MM908E624 Flash Memory: Store parameters to the LIN Slave (908EY16) Flash memory.

More information on this GUI can be found in Section 7.2.2, “GUI “Window Lift Parameters Configuration” Description”.

2.2.3 GUI “The MM908E624 Board OUTs Controlled by the LIN Bus” Introduction

This GUI introduces the MM908E624 board as a general-purpose OUTs voltage polarity and PWM controller. Offered features are as follows:

- LIN bus control:
  - Run / Stop communication
  - Sleep / Wake-up LIN Slaves
- The MM908E624 board OUTs control:
  - Board relay ON / OFF control
  - OUTs PWM control (duty cycle step approximately 0.4%)
  - Display system power supply voltage and MM908E624 device status (overvoltage, overtemperature, etc.)

More information on this GUI can be found in Section 7.2.3, “GUI “The MM908E624 OUTs Controlled by the LIN Bus” Description”.

3 Freescale Tools Used

3.1 FreeMASTER Tool

The FreeMASTER (formerly known as PC Master) software is one of the off-chip drivers, which support communication between the target microcontroller and PC. This tool allows the programmer to remotely control an application using a user-friendly graphical environment running on a PC. It also provides the ability to view some real-time application variables in both text and graphical form. It provides a lot of key features, including: Real-time debugging, Diagnostic tool, Demonstration tool, Education tool, etc.

The Window Lift application FreeMASTER tool utilization is depicted in Figure 3. Once the FreeMASTER is installed with the BDM plug-in module[1] and running on PC, it communicates with the target device via the data line using the BDM HC(S)12 multilink[2]. The FreeMASTER tool reads and writes the loaded target variables contents (target memory cells) and displays them in the GUI realized by the HTML page.

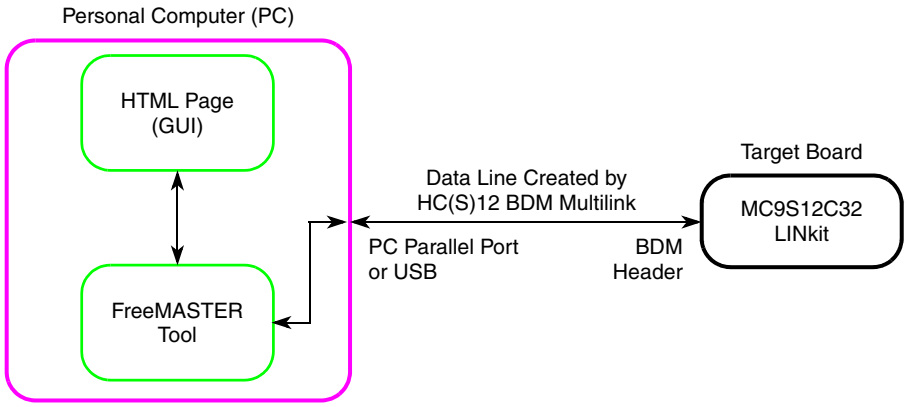


Figure 3. FreeMASTER Utilization Description

The FreeMASTER also enables the user to display loaded data in the Oscilloscope GUI component, which enables displaying real time events in a graph and in the Recorder GUI component, which is helpful during the fast speed events triggering. However, it is necessary to check if the installed plug-in module enables running the FreeMASTER oscilloscope and recorder tools also.

Freescale offers FreeMASTER tool support as follows:

- FreeMASTER — PC side application
- FreeMASTER — embedded side drivers (MCU / DSP family)
- Application notes
- Plug-in modules (RS232, BDM, JTAG, CAN)
- Application support from Freescale Semiconductor

For more information on the FreeMASTER tool, see Reference [7.].

1. Check if the required plug-in module enables running the FreeMASTER oscilloscope and recorder tools also.
2. The data line can also be realized by RS232, JTAG, and CAN, if the target board enables it.

4 Freescale Components Used

4.1 MM908E624 Integrated Triple High-Side Switch with Embedded MCU and LIN Serial Communication for Relay Drivers

The MM908E624 is an integrated single-package solution that includes a high performance HC08 microcontroller with a SMARTMOS™ analog control IC (see Figure 4).

The HC08 includes:

- Flash memory
- Timers (TIMA, TIMB)
- Enhanced serial communications interface (ESCI)
- Analog-to-digital converter (ADC)
- Serial peripheral interface (SPI) (only internal)
- An internal clock generator module (ICG).

For more information on the MCU, see Reference [2.].

The analog control die provides:

- Three high-side outputs with diagnostic functions
- Voltage regulator
- Window watchdog
- Operational amplifier
- Local interconnect network (LIN) physical layer

The single-package solution, together with LIN, provides optimal application performance adjustments and a space-saving PCB design. It is well suited to the control of automotive high-current motors applications using relays (e.g., window lifts, sun roofs, fans).

The MM908E624 features:

- High-Performance MC68HC908EY16 MCU:
  - 15872 Bytes of On-Chip Flash Memory with in-circuit programming,
  - 512 Bytes of On-Chip RAM,
  - Internal Clock Generator Module (ICG) with a trimming capability of better than 1 percent,
  - Two 16-bit, 2-channel timer (TIMA and TIMB) interface modules with selectable input capture, output compare, and pulse-width modulation (PWM) capability on each channel,
  - Timebase Module (TBM),
  - 8-channel, 10-bit successive approximation analog-to-digital converter (ADC),

  - Enhanced serial communications interface module (ESCI) suited to Local Interconnect Network (LIN) connectivity,
  - Serial peripheral interface (SPI),
  - 5-bit keyboard interrupt (KBI) with wake-up feature,
  - Low Voltage Inhibit module (LVI),
  - Computer Operating Properly Module (COP).
- SMARTMOS^{TM} analog control IC:
  - LIN Physical Layer,
  - Low Drop Voltage Regulator,
  - Operational Amplifier,
  - Window Watchdog,
  - Three High-Side Outputs,
  - Two Wake-Up Inputs.

The MM908E624 SPI and ESCI modules are utilized for communication between the MCU die and the Analog die. The SPI module controls the Analog die. The LIN physical layer, which is a part of the Analog die, is controlled by the MCU ESCI module to allow device LIN connectivity.

For more information on the MM908E624, see Reference [1.], Reference [2.], and Reference [9.].

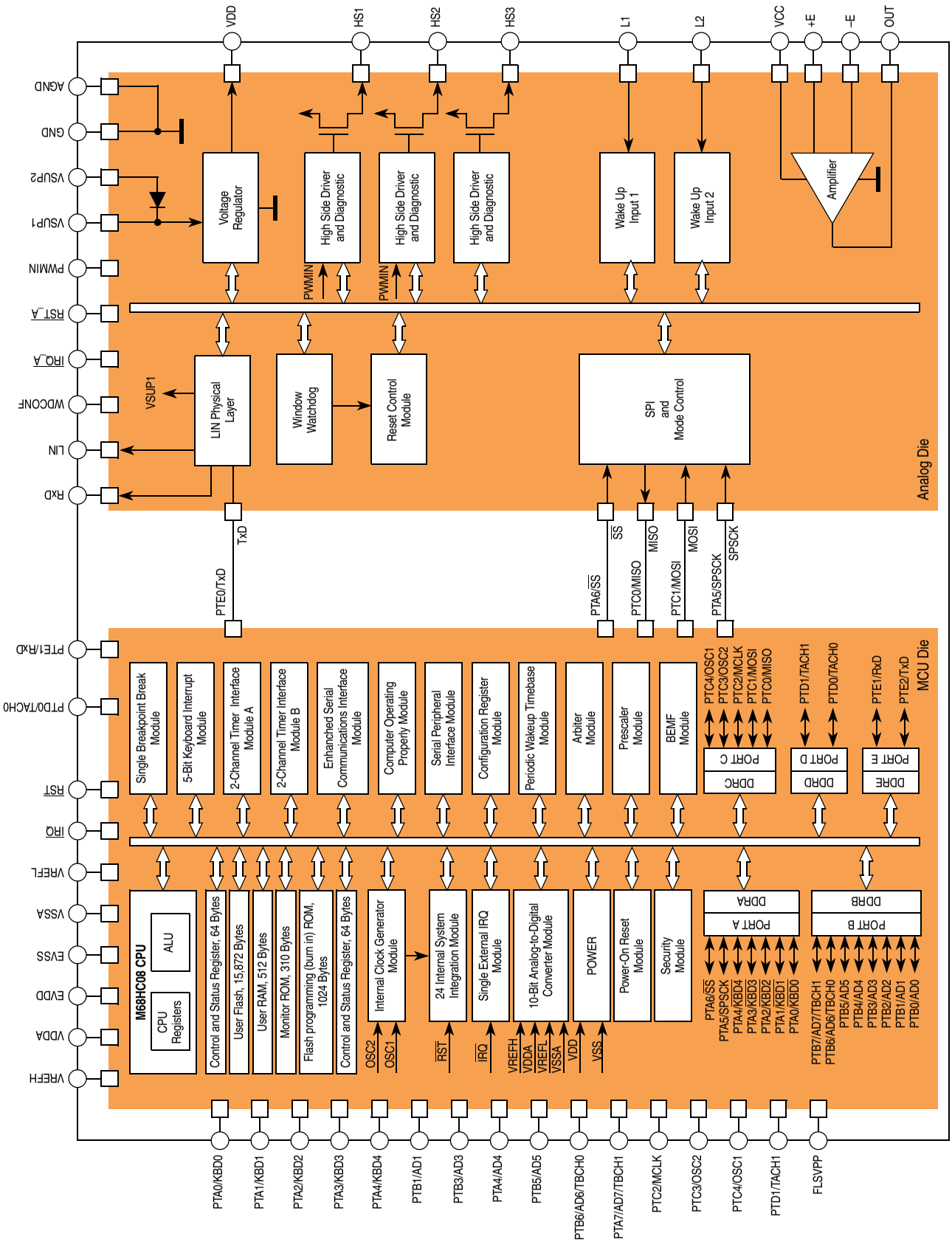


Figure 4. MM908E624 Block Diagram

4.2 MC9S12C32 16-Bit Microcontroller Unit

The MC9S12C32 (see Figure 5) is a 48/52/80-pin Flash-based Industrial / Automotive network control MCU, comprised of standard on-chip peripherals including a 16-bit central processing unit (HCS12 CPU), 32K bytes of Flash EEPROM, 2K bytes of RAM, an asynchronous serial communications interface (SCI), a serial peripheral interface (SPI), an 8-channel 16-bit timer module (TIM), a 6-channel 8-bit Pulse Width Modulator (PWM), an 8-channel, 10-bit analog-to-digital converter (ADC), and a CAN 2.0 A, B software compatible module (MSCAN). Furthermore, an on chip bandgap based voltage regulator (VREG) generates the internal digital supply voltage (VDD) for a 3 V to 5.5V external supply range. The MC9S12C32 has full 16-bit data paths throughout. The inclusion of a PLL circuit allows power consumption and performance to be adjusted to suit operational requirements. A total of 50 I/O port pins and 2 input pins are available in the 80 pin package version. Furthermore, up to 12 I/O port bits are available with Wake-Up capability from STOP or WAIT mode.

- 16-bit HCS12 core
- Wake-up interrupt inputs:
  - Up to 12-port bits available for wake up interrupt function with digital filtering
- Memory:
  - 32K Byte Flash EEPROM (erasable in 512-byte sectors)
  - 2K Byte RAM
- One Analog-to-Digital Converter 8-channel module with 10-bit resolution with an external conversion trigger capability
- One 1M bit per second, CAN 2.0 A, B software compatible modules
- 8-Channel Timer Module (TIM)
- 6 PWM channels
- Serial interfaces:
  - One asynchronous serial communications interface (SCI)
  - One synchronous serial peripheral interface (SPI)
- CRG (Clock Reset Generator Module)
- Operating frequency 25MHz Bus Speed
- Internal 2.5V Regulator:
  - Includes low voltage reset (LVR) circuitry
  - Includes low voltage interrupt (LVI) circuitry
- 48-Pin LQFP, 52-Pin LQFP, or 80-Pin QFP package:
  - Up to 58 I/O lines with 5V input and drive capability (80 pin package)
  - Up to 2 dedicated 5V input only lines (IRQ, XIRQ)
  - 5V 8 A/D converter inputs,
  - 5V I/O

For more information on the MC9S12C32, see Reference [3.].

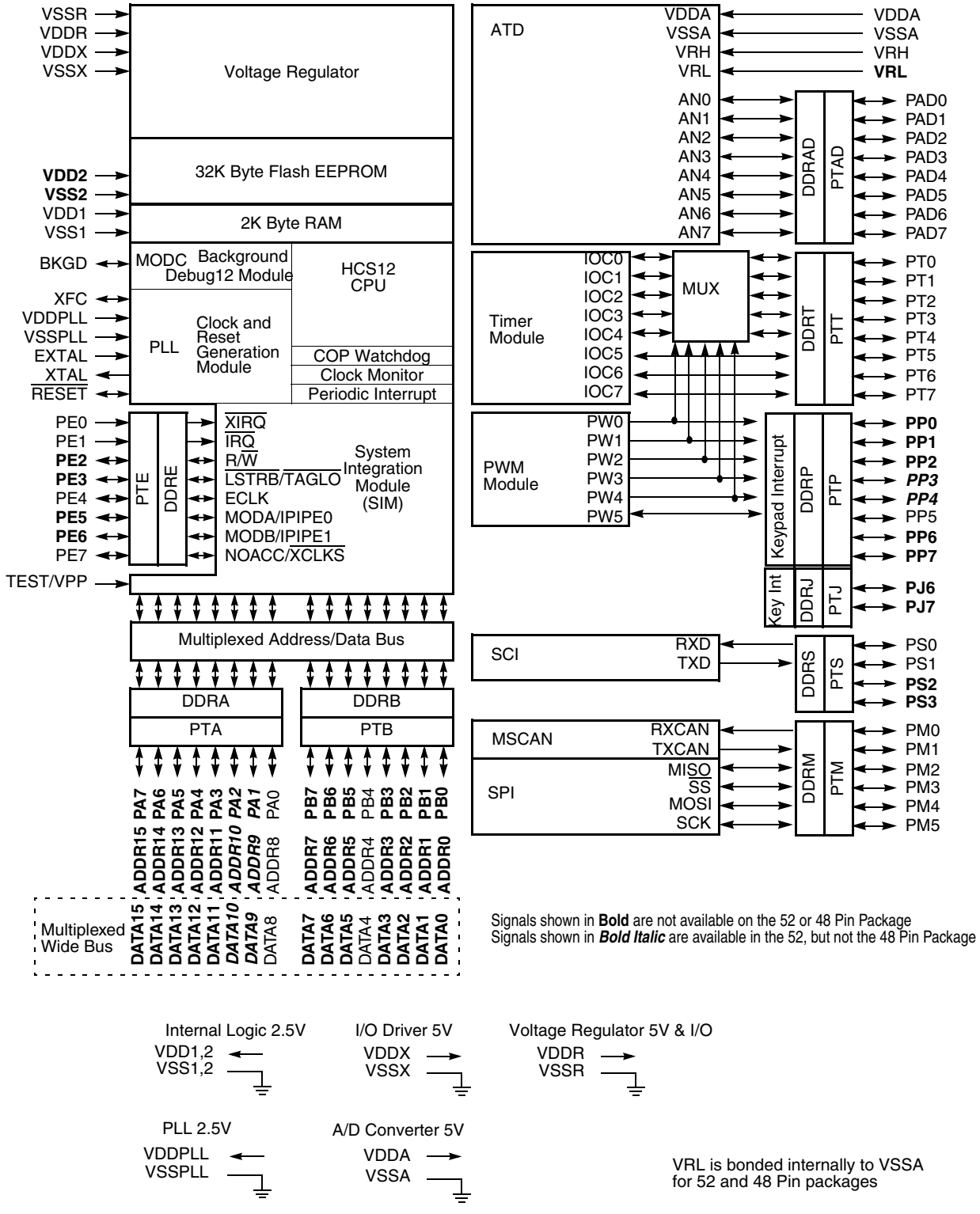


Figure 5. MC9S12C32 Block Diagram

4.3 LIN Physical Interface MC33399

This component (depicted in Figure 6) is designed for use in Master and Slave LIN nodes as a bus voltage converter with an implemented bus wake-up capability (see Reference [4.]).

Device features:

- Communication speed of up to 20kb/s
- Interfaces to the MCU with CMOS compatible I/O pins
- Two operational modes: Normal and Sleep
- Very low standby current of 20uA during Sleep mode
- An unpowered node does not disturb the LIN network
- Wake up capability from the LIN bus, MCU, or by high voltage on the wake-up pin
- Controls an external voltage regulator
- High EMC immunity

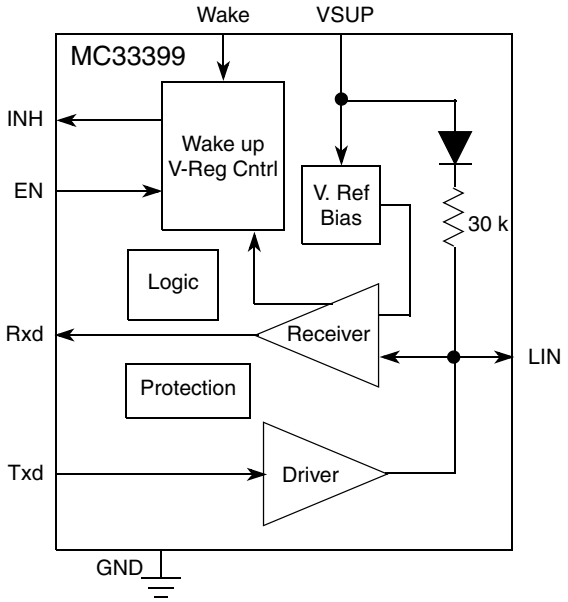


Figure 6. MC33399 (LIN Physical Interface) Block Diagram

NOTE

The new eLIN physical interface MC33661 (see Reference [5.]) fully replaces the MC33399 described above[1]. With a signal slew rate selection option, active bus signal shaping, and a special mode for operating above 100kb/s for testing and programming, it provides an excellent EMC behavior and capability for via the LIN bus Master / Slave node MCU memory programming.

1. On the LINkit Slave board, replacing the MC33399 by the MC33661 requires adding a 10kΩ resistor between the MC33661 INH pin and LT1121 /SHDN pin, because of the MC33661 higher INH pin drive capability.

5 Hardware Description

5.1 MM908E624 Board

The board “heart” (see Figure 7) is the MM908E624 single-package solution, consisting of an 8-bit MCU and an analog die with integrated LIN physical interface, power supply management, window watchdog, operational amplifier, and I/O control (see Section 4.1, “MM908E624 Integrated Triple High-Side Switch with Embedded MCU and LIN Serial Communication for Relay Drivers”).

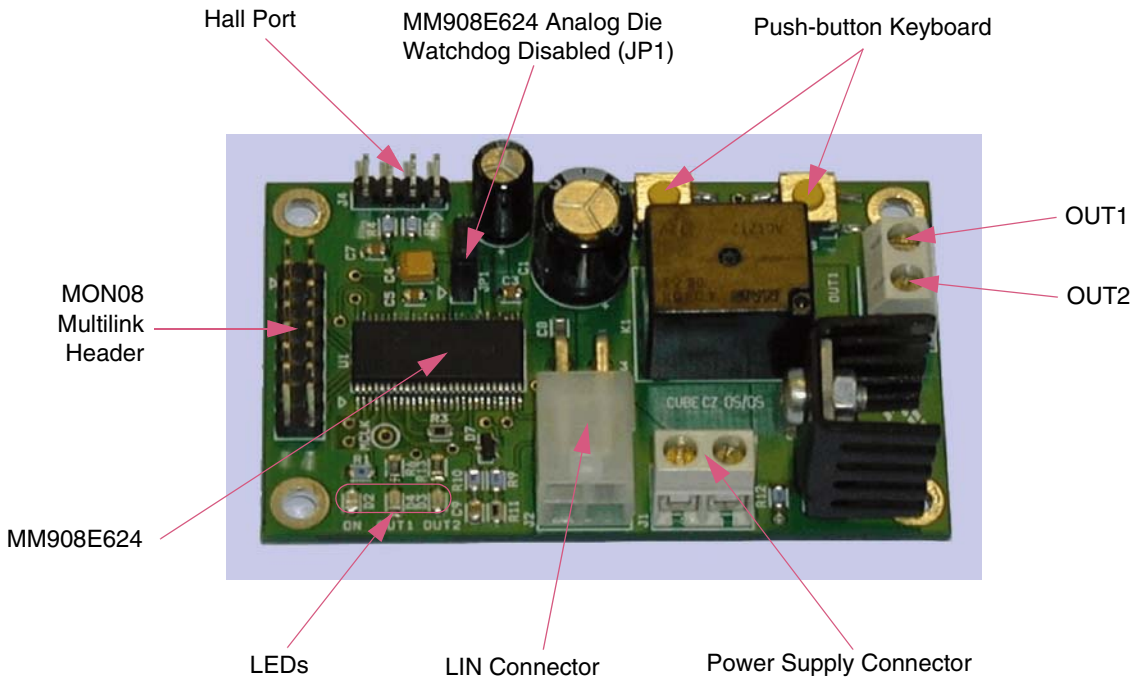


Figure 7. MM908E624 Board Description

The MM908E624 board (see also the schematic in Figure 8) was designed to enable the MM908E624 Window Lift / Sun Roof control. It offers to control the OUT1, OUT2 output voltage (+12V or GND) using the K1 relay, and to PWM control the output voltage duty cycle, realized by the Q1 MOSFET transistor. The board also provides the possibility to read three pin (voltage coded) Hall sensors for to evaluate e.g., window position or window antipinch / stall condition.

The board D2 LED displays the current MM908E624 device state. If the D2 LED is turned ON, the device is running. Otherwise, if switched OFF, the MM908E624 device is ether in Sleep mode or the device supply power is not connected.

The D3 and D4 LEDs display the board’s OUTs state. When the D3 or D4 LED is switched ON, the corresponding board OUT is supplied from the power supply. Otherwise, the OUT is connected to the power supply ground wire.

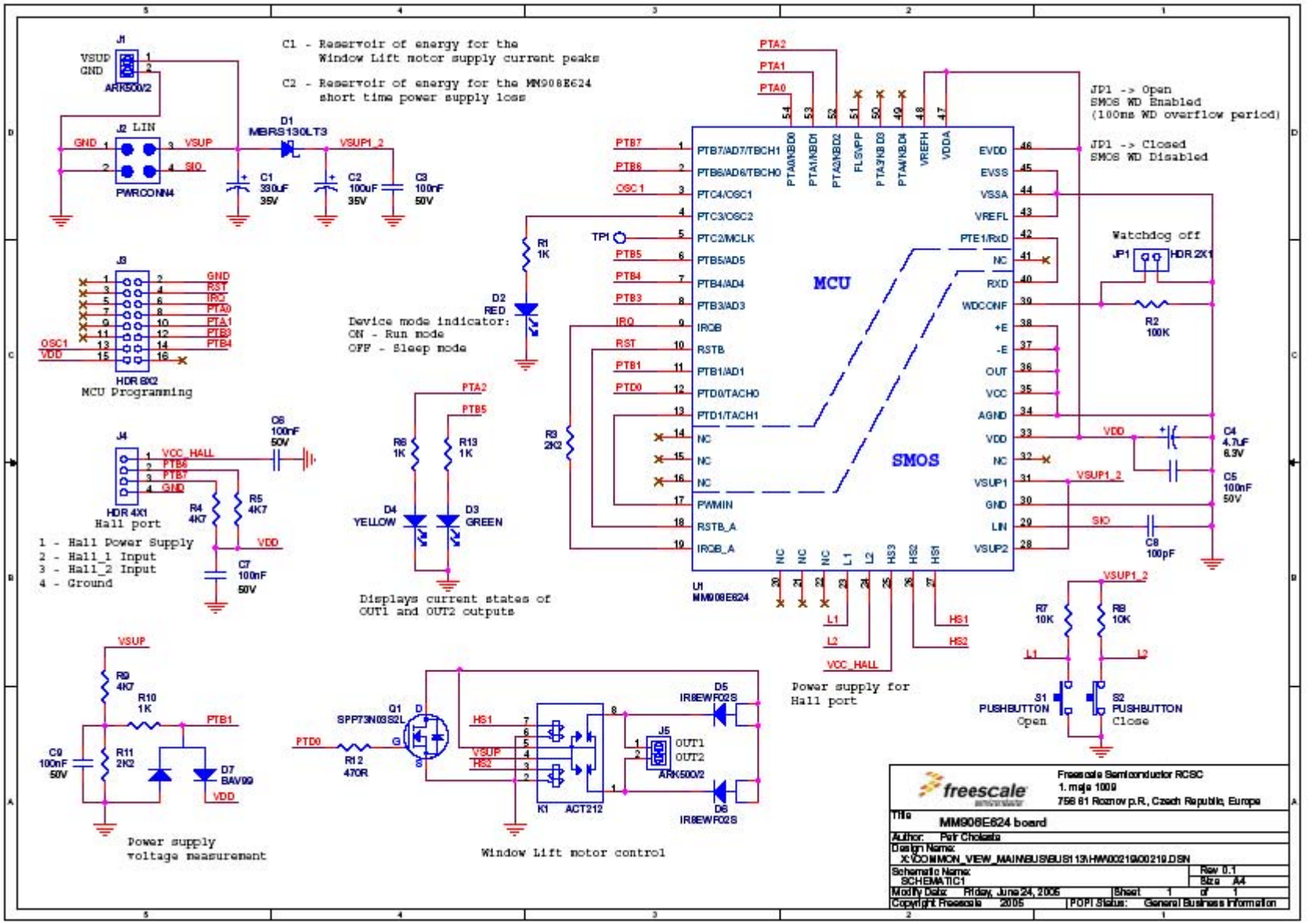


Figure 8. MM908E624 Board Schematic

The MM908E624 board interfaces with the rest of the LIN network by 4-pin LIN connector J2, which can also be used as the board power supply input. If any extra supply power is needed, the J1 connector was added to handle the high current power supply, e.g., in cases where, the LIN power supply wire is not capable of managing the supply current.

In some applications it can be useful to measure the supply voltage. For this purpose a voltage divider circuit with protection circuit (R9, R10, R11, C9 and D7) is included on the board. The measurable maximum supply voltage is 15 V.

To show the MM908E624 board “keyboard” control and device wake-up capability, a single push button keyboard, consisting of buttons S1, S2 was added (push S1 to Open Window, push S2 to Close Window).

For downloading the MCU program code, the standard 16-pin HC(S)08 multilink header J3 is included on board.

The MM908E624 board Hall Port header, LIN, and Power Supply connectors interface are depicted in Figure 9. For more information, also see Appendix B, “System Setup”.

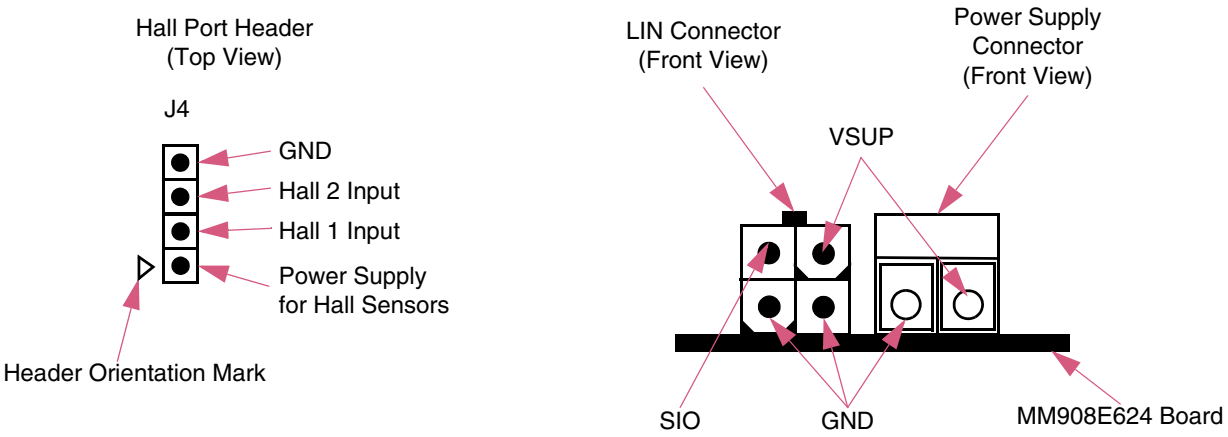


Figure 9. MM908E624 Board Connectors Interface Description

The MM908E624 board features are as follows:

- Power supply voltage range: from 5.5V to 18 V
- Power supply current[1]: cca. 22 mA (the MM908E624 device takes cca. 20 mA, the power supply measurement circuit takes cca. 2mA)
- Maximal OUTs continuous current[2]: up to 15 A
- Maximal OUTs inrush current[3]: up to 25 A

1. The typical value of the MM908E624 board power supply current at the following conditions:
   — The MM908E624 board power supply voltage equals 13.5V,
   — The board OUTs are OFF, MOSFET Q1 is OFF, Hall port is OFF, and LEDs are OFF,
   — The MM908E624 analog chip runs in normal mode,
   — The MM908E624 MCU (908EY16) is running with 20MHz internal core clock generated by ICG module,
   — The MCU ICG, TIMA, TIMB, TBM, ESCI, SPI, ADC, and I/O modules are enabled.
2. The board MOSFET Q1 turned ON, no Q1 PWM duty cycle control applied,
3. The board MOSFET Q1 turned ON, no Q1 PWM duty cycle control applied.

For the MM908E624 board OUTs PWM duty cycle control, the MOSFET Q1 power dissipation has to be calculated. The calculated value should not exceed maximal limit, otherwise the MOSFET Q1 will be destroyed.

The MOSFET Q1 maximal power dissipation calculation is based on the known value of the Q1 package thermal resistance. The Q1 type SPP73N03S2L, the thermal resistances are as follows:

- Between the chip junction and package case: R_{thjc} = 1.6K/W
- Estimated value between package case and cooler: R_{thcc} = 0.4K/W
- Between the cooler and ambient: R_{thca} = 20K/W

The Q1 MOSFET total thermal resistance R_{thQ1} [K/W] equals to sum all mentioned thermal resistances:

$$R_{thQ1} = R_{thjc} + R_{thcc} + R_{thca} = 1,6 + 0,4 + 20 = 22 \quad \textbf{Eqn. 1}$$

The Q1 MOSFET maximal power dissipation P_{Q1max} [W] can be calculated according to the following formula:

$$P_{Q1max} = \frac{T_{jmax} - T_{amb}}{R_{thQ1}} \quad \textbf{Eqn. 2}$$

where:

T_{jmax} — maximal operating temperature [°C] (175°C for SPP73N03S2L),
T_{amb} — ambient temperature [°C],
R_{thQ1} — total thermal resistance [Ω].

The Q1 MOSFET power dissipation consists mainly of resistive P_{Q1res} [W] and switching P_{Q1sw} [W] losses:

$$P_{Q1} = P_{Q1res} + P_{Q1sw} \quad \textbf{Eqn. 3}$$

The switching losses of the Q1 P_{Q1sw} [W] can be calculated by the following formula:

$$P_{Q1sw} = \frac{V_{sup} \cdot I_{load}}{2} \cdot (t_r + t_f) \cdot f_{sw} \quad \textbf{Eqn. 4}$$

where:

V_{sup} — power supply voltage [V],
I_{load} — maximal load current [A],
t_r — Q1 drain-source current rise time [s],
t_f — Q1 drain-source current fall time [s],
f_{sw} — Q1 switching frequency [Hz].

The resistive losses of the Q1 P_{Q1res} [W] can be calculated:

$$P_{Q1max} = I^2_{loadrms} \cdot R_{DSON} \quad \textit{Eqn. 5}$$

where:

$I_{loadrms}$ — rms value of load current [A],
R_{DSON} — Q1 drain to source ON resistance [Ω] at operating temperature.

The Q1 MOSFET R_{DSON} [Ω] value is dependent on the operating temperature. For the R_{DSON} re-calculation, it is possible use the following formula:

$$R_{DSON} = R_{DSONspec} \cdot (1 + 0,005 \cdot (T_{op} - T_{spec})) \quad \textit{Eqn. 6}$$

where:

$R_{DSONspec}$ — Q1 drain-source ON resistance [Ω] at temperature T_{spec} [°C] (usually 25°C),
T_{op} — Q1 operating temperature [°C].

5.2 MC9S12C32 LINkit Board

For the Window Lift LIN Master device, the MC9S12C32 LINkit board is used (see Figure 10).

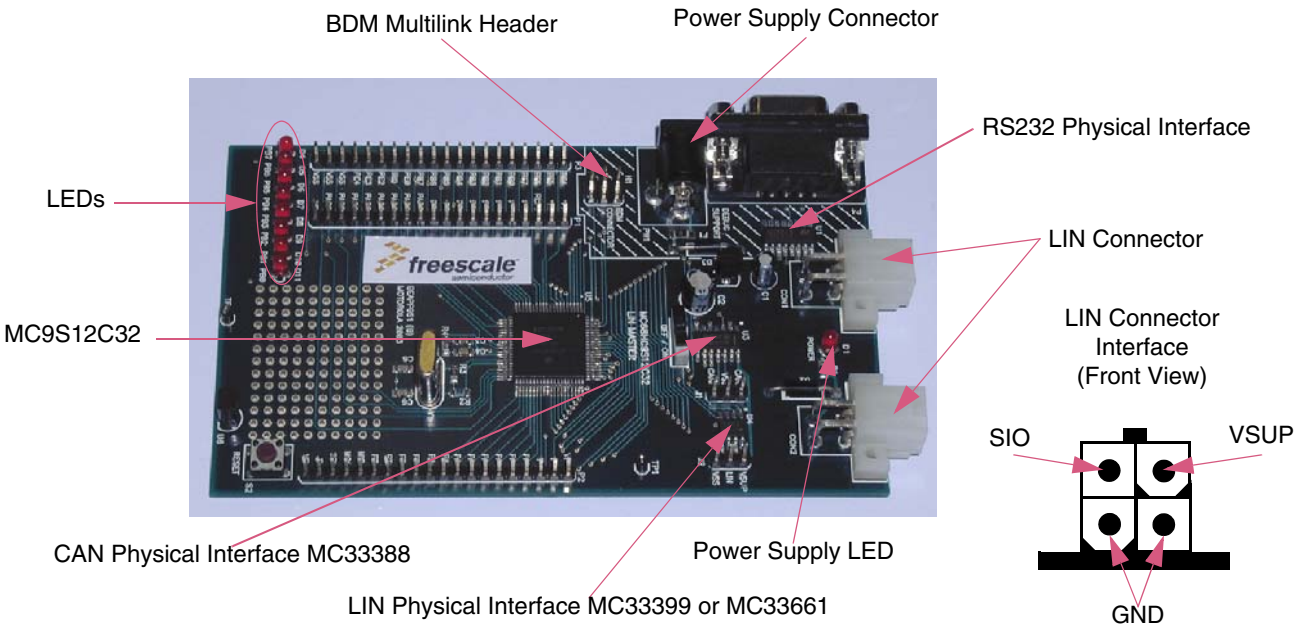


Figure 10. MC9S12C32 LINkit Board Description

The MC9S12C32 Master LINkit (see also LINkit schematic in Figure 11) consists of the 16-bit MC9S12C32 MCU, physical interfaces for CAN, RS232 and LIN connectivity, and the power supply management. The MC9S12C32 LINkit board conception is closely linked to the expectation that the LIN Master can also act as the LIN to a superior bus gateway, e.g. as in the case of the MC9S12C32 LINkit, the LIN to CAN gateway.

The MC34064 device is also included on the LINkit board, to secure the proper MCU start after a low power voltage condition has occurred. To obtain more information on this topic, see Reference [12.].

The MC9S12C32 LINkit was designed to be capable of the LIN network power supply. Let's consider that the power supply current of the LINkit board with the BDM multilink connected is about 200mA and the on board fuse F1 is limiting the power supply current to 500mA (see schematic in Figure 11). The remaining LINkit board power supply capability is about 300mA. This enables running an application only in demonstration mode without any real Window Lift system connection. In order to show this board as the real Window Lift LIN Master, the board is supplied from the MM908E624 board by the LIN bus. The MM908E624 board includes an extra connector, J1 (see schematic in Figure 8), for the high power supply current management.

For downloading the MCU program code, a standard 6-pin HC(S)12 BDM multilink header is included on board.

5.3 Car Door Window Lift Platform

The Window Lift application was developed for the control of a Window Lift driven by a 12V DC motor and controlled by two separate three pin Hall sensors. The Hall pulse period generated by the Car Door Window Lift platform used, corresponds to a window glass shift of 1mm.

The Window Lift application uses Hall sensor signals for:

- Window movement direction control (Open / Close)
- The actual window position reading
- Window glass antipinch and stall detection

To obtain more information on system setup, see Appendix B, "System Setup".

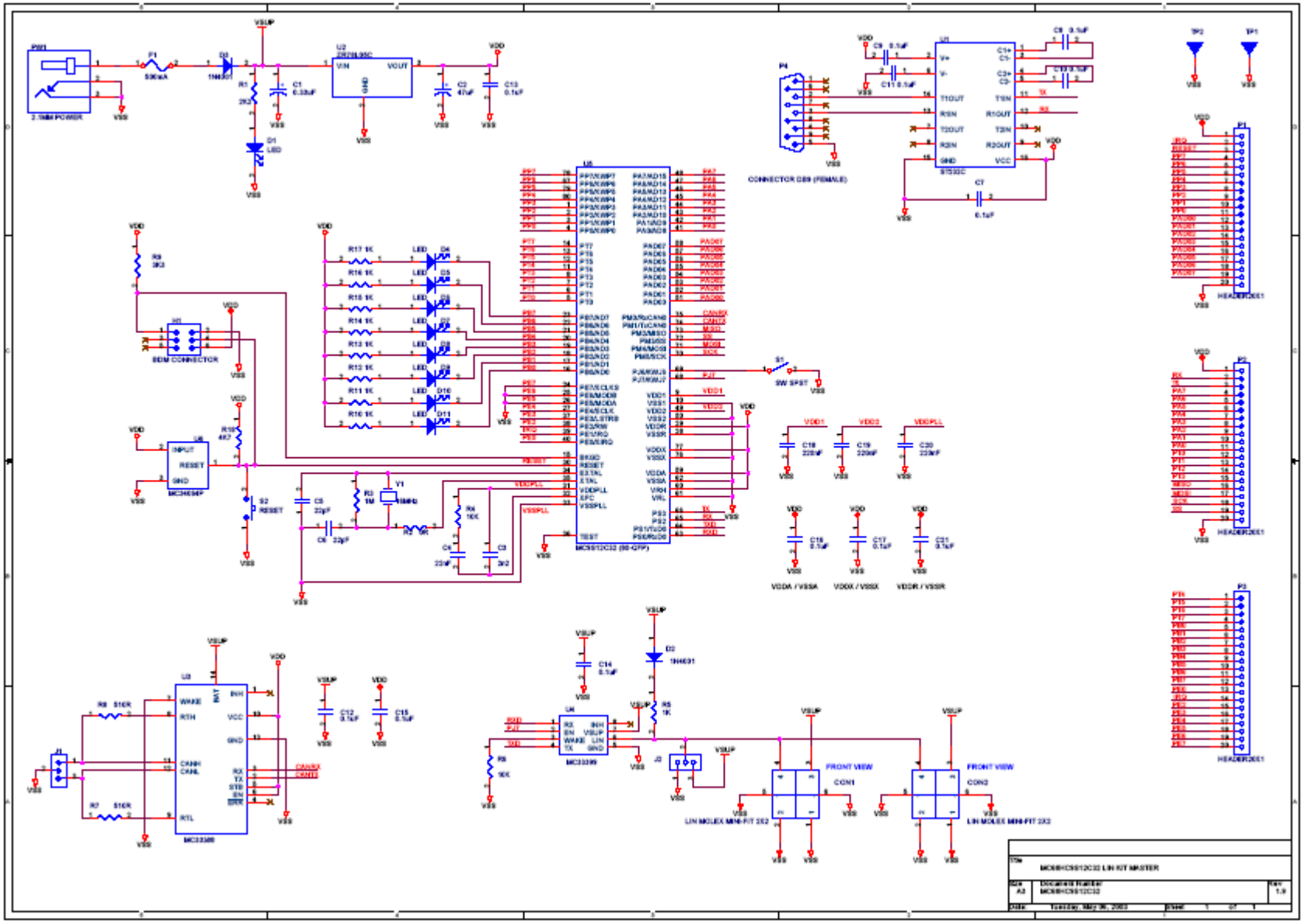


Figure 11. MC9S12C32 LINkit Board Schematic

6 Software Description

6.1 LIN Slave Software Arrangement

6.1.1 General Description

The LIN Slave software can be separated into two main parts. The first covers the LIN connectivity related routines, the second covers the application itself. Figure 12 displays how the project folders are arranged in the Metrowerks CodeWarrior Stationary.

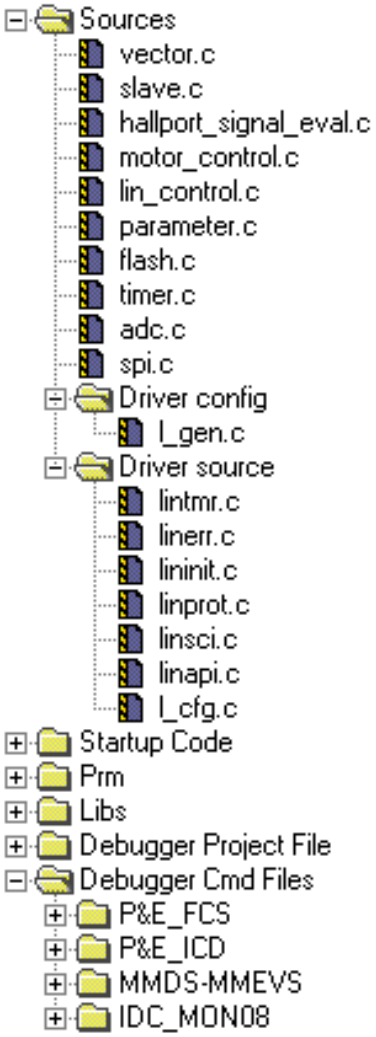


Figure 12. LIN Slave Metrowerks CodeWarrior Stationary

6.1.2 LIN Connectivity Related Routines

The LIN connectivity software covers all LIN communication (see Appendix A, “Messaging Strategy”). The application controls LIN communication by the receiver and the transmitter signal command buffers and flags. The LIN communication runs at 9.6 kBd.

The LIN connectivity is implemented using the Freescale LIN 1.3 driver software package, which is available free of charge on the Freescale web pages detailed in Reference [11.].

In the project directory, the Freescale LIN 1.3 driver package is represented by folders as follows:

- Driver configuration: This folder includes the l_gen.c file that together with the l_gen.h file interfaces the LIN driver to an application. Those files are written in LIN API and user accessible to allow LIN network modification,
- Driver source: This folder includes LIN1.3 driver source files.

6.1.3 Application Software

The application software controls the Car Door Window Lift platform according to the LIN Master node commands (see Appendix A, “Messaging Strategy”).

The functions provided are as follows:

- Moving the window glass:
  - Close / open,
  - Close completely / open completely,
  - Reach desired position,
  - Enable control of window glass position by the MM908E624 board push button keyboard,
  - Window soft start / soft stop (done by MOSFET Q1 OUTs PWM ramp control).
- Reporting current window status:
  - Closed completely / opened completely,
  - Closing / opening,
  - Stopped,
  - Window reached desired position and it was stopped (realized by Hall sensors signal evaluation),
  - Window antipinch / stall occurred and window was stopped,
  - Window position (measured in Hall signal pulse half periods),
  - Hall signal half pulse period measurement.
- Reporting MM908E624 board status:
  - Board keyboard status (LIN Slave requirement to Open / Close / Stop window movement)
  - Board power supply measurement,
  - MM908E624 device report (High Sides over-temperature, Voltage Regulator over-temperature, Power supply Low / High Voltage, MM908E624 LIN layer overcurrent / overvoltage).

The application software can be run in three different modes:

1. Normal mode (see Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction” and Section 7.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Description”),
2. Keep Window Speed mode (see Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction” and Section 7.2.2, “GUI “Window Lift Parameters Configuration” Description”),
3. PWM Control mode (see Section 2.2.3, “GUI “The MM908E624 Board OUTs Controlled by the LIN Bus” Introduction” and Section 7.2.3, “GUI “The MM908E624 OUTs Controlled by the LIN Bus” Description”).

The application software is represented in the project directory by the files:

- Sources:
  - slave.c and slave.h (main application files),
  - target.h (target dependent stuff),
  - hallport_signal_eval.c and hallport_signal_eval.h (related to the Hall port signals evaluation),
  - motor_control.c and motor_control.h (Window Lift DC motor control - rotation direction and speed control),
  - lin_control.c and lin_control.h (LIN connectivity related files),
  - parameter.c and parameter.h (system parameters Flash store / read related routines),
  - flash.c and flash .h (Flash store and read execution related routines),
  - timer.c and timer.h (TIMA, TIMB, TBM initialization),
  - adc.c and adc.h (ADC conversion related routines),
  - spi.c and spi.h (SPI related routines).

6.1.4 Application Software Configuration Files

The application software is linked with the MM908E624 hardware by *target.c* file. This file includes macro declaration that determines the behavior of the Window Lift / Sun Roof application. The most important macros are:

- ANTIPINCH_OR_STALL_THR: It defines the number of successful antipinch / stall detection cycles to arise an application window antipinch or stall evaluation process. One antipinch / stall detection cycle is executed if the measured difference of two following Hall1 pulse periods exceed the limit value preset in the system parameter structure (called sParameter.tickDifThr). The parameter value can also be updated by means of the GUI described in Section 2.2.2, “GUI “Window Lift Parameters Configuration” Introduction”, *Assign an Antipinch Threshold* variable. A value entered equals the number of TIMB ticks,
- STALL_THR: It defines the zone of window stall detection to enable window control, if the window is completely opened or closed. The value is entered as a number of TIMB channel0 interrupts. During the interrupt code execution, the window glass position counter is also incremented / decremented,

- PWM_DUTY_MIN_STOP: It equals the minimal OUTs PWM duty cycle, when a window movement soft stop command is executed,
- PWM_DUTY_MIN_POS: It defines the minimal OUTs PWM duty cycle when a window go to desired position command is executed and the window position is close to a window soft stop execution. The value entered is the OUTs PWM duty cycle in which the window reaches the desired position,
- PWM_DUTY_MAX: The maximal OUTs PWM duty cycle during the window soft start command execution. When the OUTs PWM ramp duty cycle reaches this maximal value, it is automatically set to 100% to enable running the window with full platform mechanism strength,
- PWM_RAMPSTEP: It determines the OUTs PWM ramp step,
- PWM_SPEEDSTEP: This is the PWM duty cycle step during the MM908E624 board “Keep Window Speed” mode execution, which enables keeping the window movement speed independent to the power supply voltage variation,
- PWM_DUTY_UPDATE: It defines the timebase of OUTs PWM duty cycle updates during device “Keep Window Speed” mode execution. The least timebase period step equals the TBM module interrupt period execution,
- TICK_TO_PWM_START: It determines the number of TIMB channel0 interrupts (equals to window position counts) when, during a soft stop window or go to desired position command execution, the OUTs PWM duty cycle ramp generation will be started,
- TICK_WIDTH: It equals a Hall1 pulse period that the application software keeps constant by the OUTs PWM duty cycle control during a device “Keep Window Speed” MM908E624 mode execution,
- HALL_PORT_NONE_SIGNAL: Number of TBM arrised interrupts when “None Signal on Hall port” is detected,
- TIME_TO_START_ANTIPINCH: Number of TBM arrised interrupts when the window antipinch / stall detection execution is enabled,
- COUNTS_TO_START_ANTIPINCH: Number of TIMB channel0 interrupts (equals to window position counts), when the window antipinch / stall detection execution will be started.

The MM908E624 LIN connectivity can be configured in the *lin_control.h* file. The defines are as follows:

- LIN_WL_SLAVE_LEFT[1]: By defining this symbol, the MM908E624 node acts like a Left Window Lift Slave node,
- LIN_WL_SLAVE_RIGHT [2]: By defining this symbol, the MM908E624 node acts like a Right Window Lift Slave node,
- ARBITER_DATA_USED: The ESCI arbiter data are used for the difference between the current and desired MCU bus clock frequency calculation. The operation result determines the system variables correction result. This macro always has to be defined to allow proper Window Lift system functionality,

1. The LIN_WL_SLAVE_LEFT define must be commented, if the MM908E624 board should acts as the Right Window Lift LIN Slave node.
2. The MM908E624 board can perform either as a Right or Left Window Lift Slave node. Both nodes are not allowed to run on one MM908E624 board at the same time.

- CHECKSUM_OVER_THE_ID: Uncomment this enables calculating LIN frames checksum over the frame identifier and data, to keep the frame structure LIN 2.0 Specification compliant. However, it is also necessary to mention that the LIN Slave software is not LIN 2.0 complaint because the node configuration, which is mandatory for a LIN 2.0 devices, is not implemented (see Reference [13.]). The MM908E624 board can act as a LIN 2.0 compliant device, if the LIN 2.0 drivers are implemented in the node (see Reference [14.] and Reference [15.]),
- EY16_BREAKDELIMITER_WORKAROUND: Uncomment this enables the MC68HC908EY16 LIN Break Delimiter Recognition issue workaround. The LIN Break Delimiter Recognition issue can arise when the LIN Slave ESCI clock is slower than the LIN Master SCI clock, the break delimiter signal ("1") is just one bit long and the break signal ("0") is 1x.5 bit long (for more information see Reference [16.]). This define should always be uncommented to enable proper LIN 1.3 driver functionality.

6.2 LIN Slave Software Description

6.2.1 Application Software Main Code

An application main loop software is the part of the application code written in file *slave.c*. The main loop flow chart is depicted in Figure 13.

The main application code consist of two independent blocks. The first code block serves the MM908E624 board initialization, the second code block is represented by the main code loop, which controls the MM908E624 board LIN connectivity and an application performance.

The LIN connectivity is realized by the LIN 1.3 Freescale driver for HC(S)08 microcontrollers, which is downloadable from the Freescale web pages (see Reference [11.]) free of charge. As an example of LIN 1.3 driver usage, the LIN API driver software interface was chosen for the LIN Slave node application utilization. To introduce also the other LIN1.3 driver software interface utilization, the LIN Master node application uses the Freescale API. To obtain more information on application LIN connectivity, see Appendix A, “Messaging Strategy” also.

To keep the LIN Slave node functionality compliant to the LIN 1.3 Specification (see Reference [6.]), the MM908E624 board enters the low power mode (Sleep mode), if that is required by the LIN Master or if a no LIN bus activity 2.6sec[1]. period has expired. The LIN Slave can be woken up either by the LIN Master, issuing a LIN Wake-up frame, or by the action of the MM908E624 board push button keyboard, which causes the MM908E624 device to wake-up with a consequential LIN bus Wake-up frame issue.

The MM908E624 LIN Slave board provides the LIN Master with information on power supply voltage and MM908E624 analog die status (over-voltage, over-temperature, etc.) reports.

1. The LIN1.3 Protocol Specification (see Reference [6.]) assigns the LIN bus Idle timeout as 25000 of T_{bit} time. An application LIN communication runs at 9.6KBd, i.e. $T_{bit} = 1/ f_{BR} = 1 / 9600 = 104us$. The LIN bus timeout $T_{out} = 25000 * 104us = 2.6sec$.

During the main loop execution system variables (e.g. the antipinch threshold) are also re-calculated, to keep the Window Lift application independent of the MM908E624 MCU (EY16) bus clock frequency. The source of the bus clock, Internal Clock Generator (ICG), runs within the frequency tolerance +/-25%. The LIN connectivity correct functionality in so wide a bus clock frequency margin is enabled by using the ESCI module with a fine adjust prescaler and an arbiter module, which it is used as the LIN bit time period measurement unit during LIN frame synchronization field reception (see Reference [6.]). The currently measured LIN bit time period is processed by the Window Lift application software to obtain the difference value between the desired and running bus clock frequency. The calculation result is used as the reference value for the system variables re-calculation.

The MM908E624 board was designed to run the Window Lift and the Sun Roof applications, that use the Hall sensors control. Those device utilizations are covered by application software running in the following modes:

- Normal,
- Keep Window Speed.

To obtain more information on the MM908E624 board performance running in Normal mode, see Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction”, Section 7.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Description”, and Section 6.2.3, “Application Software Stream During Device “Normal” Mode”. The MM908E624 board performance during device “Keep Window Speed” mode is represented by chapters Section 2.2.2, “GUI “Window Lift Parameters Configuration” Introduction”, Section 7.2.2, “GUI “Window Lift Parameters Configuration” Description”, and Section 6.2.4, “Application Software Stream During Device “Keep Window Speed” Mode” as well.

The MM908E624 single-package solution can also be used in each application where OUTs relay control, with or without OUTs PWM control, is required. To introduce this device capability, the MM908E624 board software can also be run in mode:

- PWM Control.

To obtain detailed information on the MM908E624 board functionality, see Section 2.2.3, “GUI “The MM908E624 Board OUTs Controlled by the LIN Bus” Introduction”, Section 7.2.3, “GUI “The MM908E624 OUTs Controlled by the LIN Bus” Description”, and Section 6.2.5, “Application Software Stream During Device “PWM Control” Mode” as well.

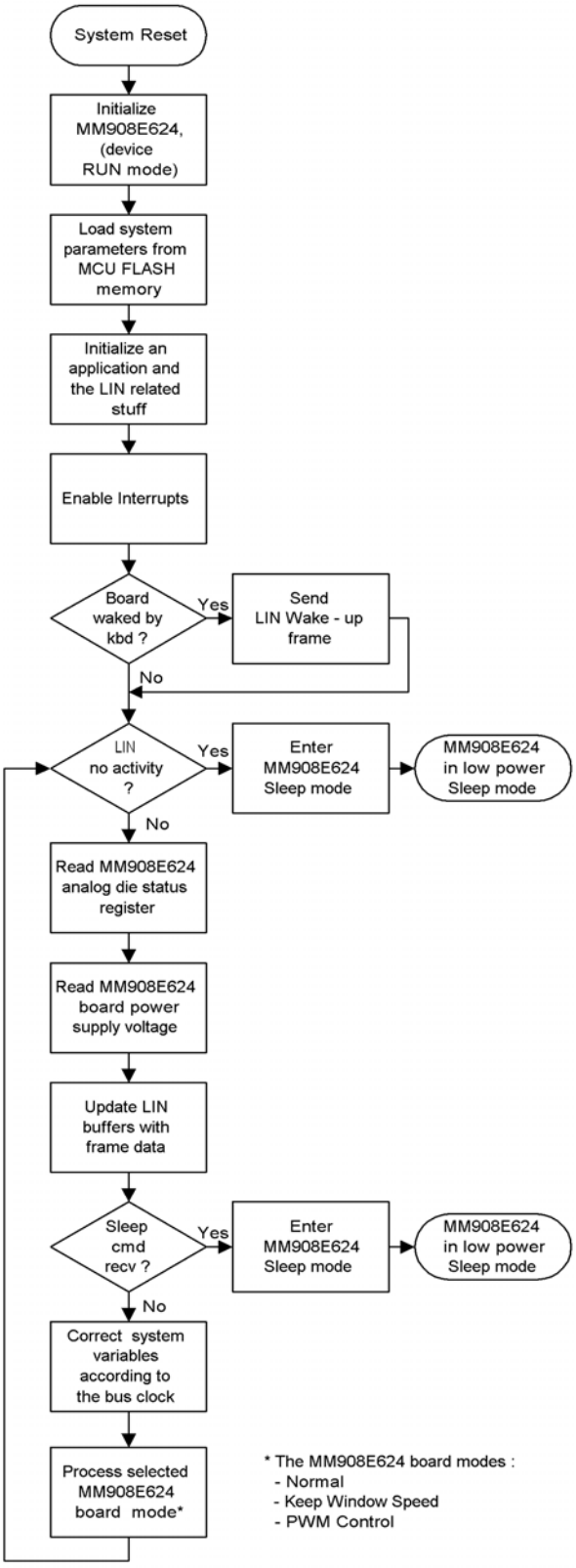


Figure 13. LIN Slave Main Loop Flow Chart

6.2.2 Application Interrupts

There are four sources of the MCU interrupts (see Figure 14):

1. TBM isr: Timebase module interrupt acts as an application timebase source. During the interrupt code execution, the LIN bus no activity counter is updated to enable the device to enter sleep mode. It also updates the current board OUTs PWM duty cycle, if either a window soft start or soft stop is processed. During a window movement start, the window antipinch is disabled for the last 4mm of window movement. To protect the Window Lift / Sun Roof application against damage, if for some reason the antipinch is not enabled after 4mm of window movement, a period of approximately 170ms antipinch disable is added and is periodically updated during TBM interrupt service routine processing. The window stall detection is based on the Hall port signal control and last window glass position evaluation,
2. ESCI isr: ESCI receive and error interrupt is occupied by Freescale LIN 1.3 driver to run the application LIN connectivity. During an interrupt execution, the LIN bus no activity counter is loaded with an initial value to postpone the device sleep mode entry,
3. TIMB channel0 isr: TIMB channel0 interrupt is captured on a Hall1 sensor signal pulse falling or rising edge. During an interrupt code execution, the window position counter is updated, the Hall2 signal state is read to evaluate the window movement direction, the TIMB channel 0 register is read, whose value is used for the window antipinch / stall detection and the Hall1 pulse period measurement (necessary for correct window movement speed control during device “Keep Window Speed” mode execution),
4. IRQ isr: the MCU external interrupt connects the MM908E624 analog die with the MCU to provide information on Analog die faults. The IRQ arises, if any Analog die hardware error occurs (over-voltage, over-temperature, etc.). During the interrupt request serving, all activities are stopped and system parameters are stored into MCU Flash memory as a preventative action before a possible system failure.

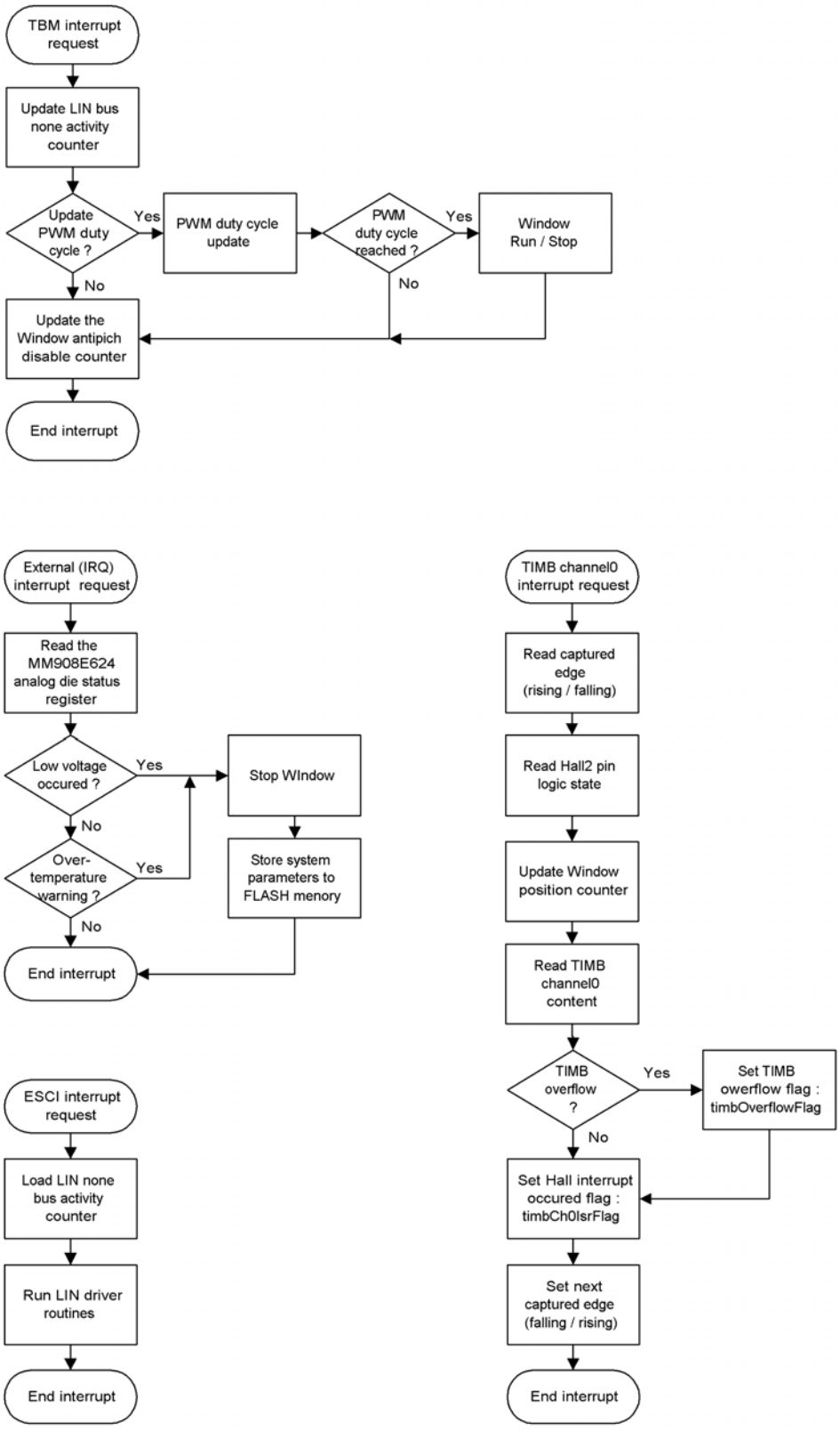


Figure 14. LIN Slave System Interrupts Flow Chart

6.2.3 Application Software Stream During Device “Normal” Mode

The MM908E624 board “Normal” mode represents standard MM908E624 utilization. This mode offers the user, by single LIN command, control of the Window Lift / Sun Roof window glass position with or without the soft start / soft stop control, which is done by the OUTs PWM ramp processing. The OUTs ramp control during window soft start / soft stop helps to reduce system electromagnetic emissions and also enables starting and stopping the window more fluently. The application features detailed description can be found in Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction” and Section 7.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Description”.

The “Normal” mode application software controls the board OUTs voltage polarity, to enable control of Window LIft / Sun Roof motor rotation in both directions. This control is realized by switching relay K1 contacts (see schematic in Figure 8). The MOSFET Q1 is used for OUTs PWM ramp duty cycle control during window glass soft start / soft stop processing. The ramp is initialized, when a new window control command is received by the LIN bus with the OUTs ramp control request bit set. If no window soft start / soft stop is required, the Q1 is turned ON.

The OUTs PWM ramp processing is controlled by the TBM interrupt service routine. During TBM interrupt service routine execution, the current OUTs PWM duty cycle is evaluated. If the PWM duty cycle equals a preset, PWM ramp update is stopped and the OUTs are either ON, if the last received window command requires a window run, or OFF, if the last command requires a window stop.

When moving the window, the Hall port signals are processed to enable the window glass movement direction, window glass position, and window glass antipinch / stall control. The Hall port signal evaluation is based on the edge triggered TIMB channel0 interrupt execution, which it is described in Section 6.2.2, “Application Interrupts”. The window glass antipinch / stall detection is realized by the following Hall signal periods difference calculation. This calculated difference is compared to a threshold limit (see Section 2.2.2, “GUI “Window Lift Parameters Configuration” Introduction” and Section 7.2.2, “GUI “Window Lift Parameters Configuration” Description”, parameter Assign an Antipinch Threshold), which can be adjusted by an application user to enable window glass antipinch / stall pressure control. The window glass antipinch / stall control is disabled for the last 4mm of window glass movement during start and stop processing to enable a fully close window.

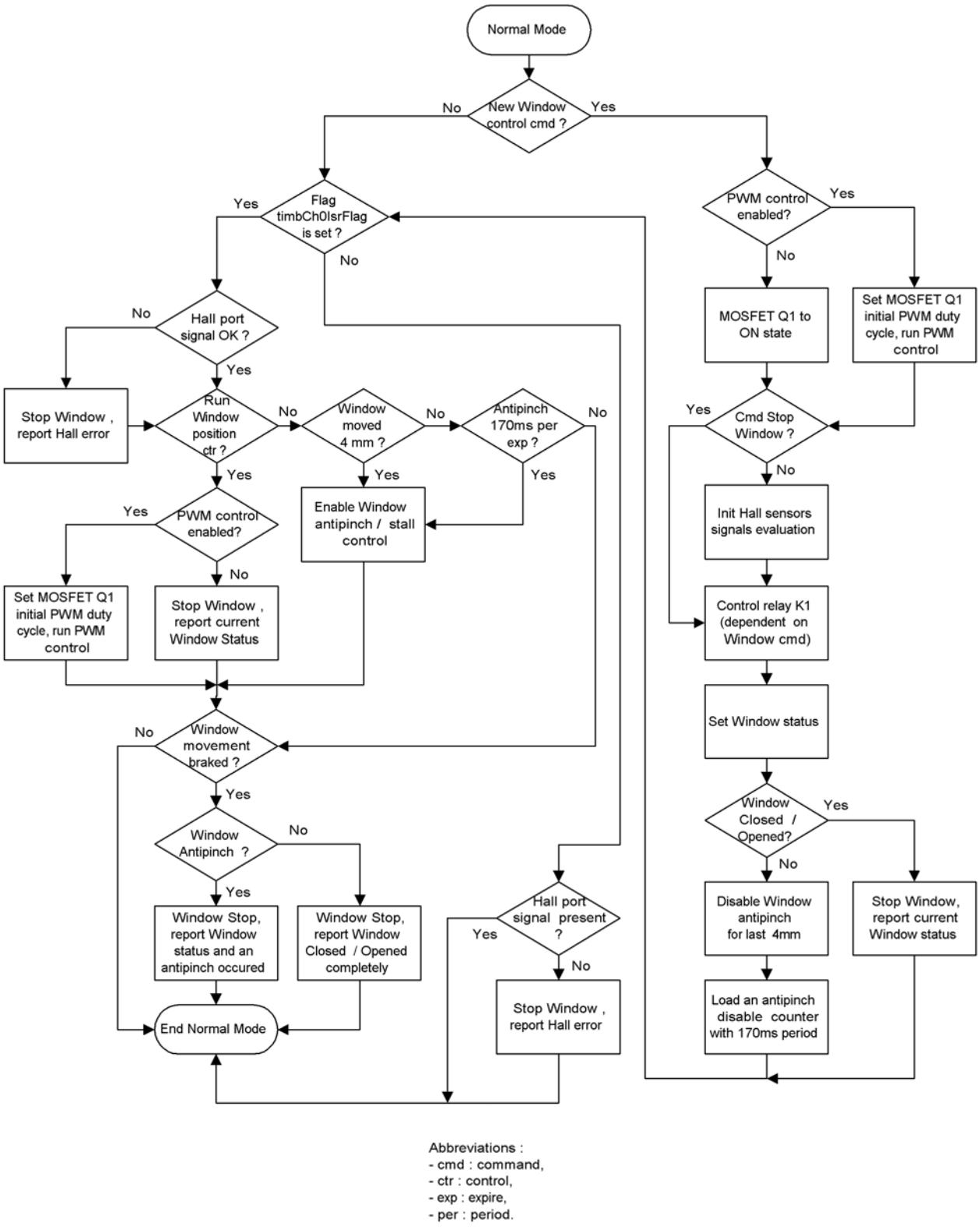


Figure 15. LIN Slave “Normal” Mode Flow Chart

6.2.4 Application Software Stream During Device “Keep Window Speed” Mode

The “Keep Window Speed” mode was added to the Window Lift / Sun Roof application to show the possibility of Window Lift / Sun Roof window movement speed control, which in this case makes the window glass movement speed independent of the power supply voltage. This process enables the Hall1 pulse period measurement, which is done by an edge triggered TIMB channel0 interrupt execution (see Section 6.2.2, “Application Interrupts”). The measured period is compared with the one predefined. The comparison result determines, if the OUTs PWM duty cycle will be increased or decreased. This process observes the time schedule based on the TBM interrupt execution. The OUTs PWM duty cycle update is disabled when the possibility of a window antipinch / stall arises (see Figure 16).

The MM908E624 board “Keep Window Speed” mode offers the user analogous Window Lift control as in the “Normal” mode. From an application user point of view, there is only one difference in application control, in that the window glass sort start / soft stop can not be disabled. This links to the “Keep Window Speed” application utilization in the High-End Window Lifts, where the soft start / soft stop feature can be taken as an obvious window glass start / stop process execution. The window soft start / soft stop process helps to reduce system EMC emission and the window glass movement start / stop is also more fluent than in “Normal” mode.

As in the case of MM908E624 board “Normal” mode, antipinch / stall detection is disabled for the last 4mm of window glass movement in window start / stop process execution to enable closing the window completely.

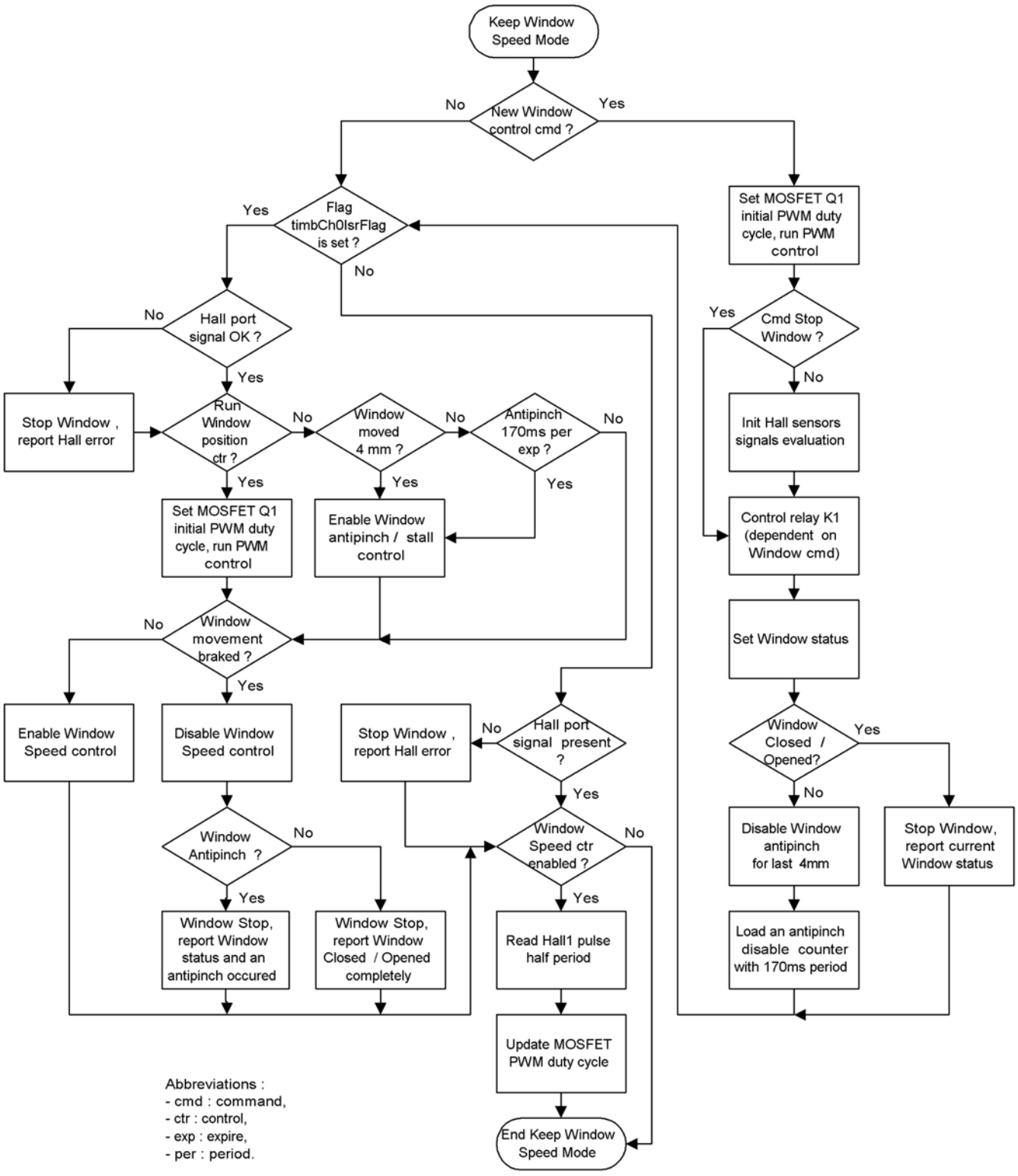


Figure 16. LIN Slave "Keep Window Speed" Mode Flow Chart

6.2.5 Application Software Stream During Device “PWM Control” Mode

The MM908E624 board “PWM Control” mode introduces the MM908E624 as a general purpose relay driver. It controls the board OUTs voltage polarity by relay K1 (see Figure 8) and also the OUTs PWM duty cycle, if required. This enables using the MM908E624 in each application where it is necessary to control high current loads or where the OUTs relay control is from some reason suitable (e.g., control circuit and load circuits galvanic separation is needed).

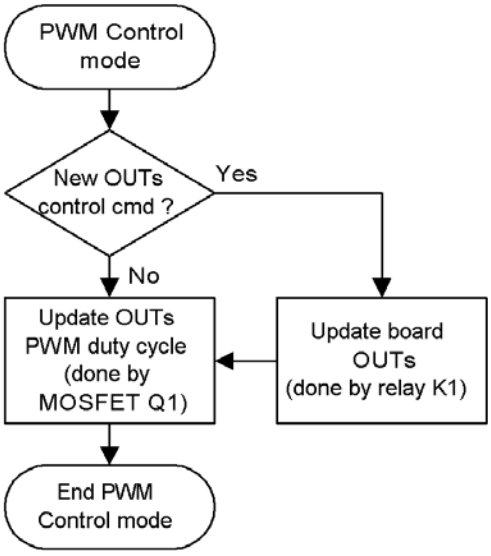


Figure 17. LIN Slave “PWM Control” Mode Flow Chart

6.3 LIN Master Software Arrangement

The LIN Master Software can be separated in two main parts. The first part covers the Master node LIN connectivity, the second part interfaces the LIN communication schedule and the contents of frames issued to the FreeMASTER tool. Both those parts create one unit capable of allowing the user to control the LIN Master node connectivity by intelligent GUIs running in the FreeMASTER tool (see Section 2.2, “System Features”).

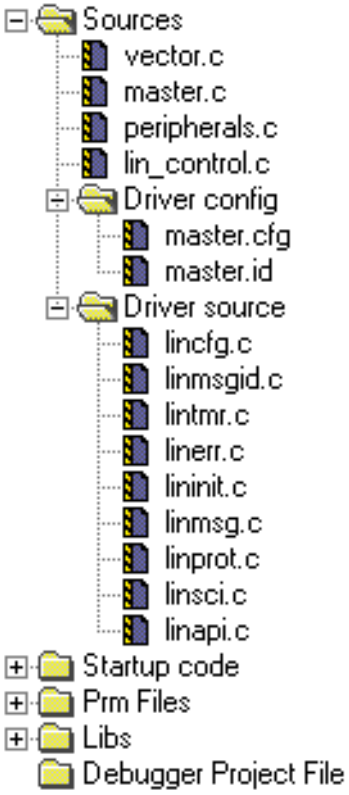


Figure 18. LIN Master Metrowerks CodeWarrior Stationary

6.3.1 LIN Connectivity Related Routines

The LIN connectivity software covers all LIN communication (see Appendix A, “Messaging Strategy”). The Window Lift application controls LIN communication by the receiver and the transmitter signal command buffers and flags. The LIN connectivity runs at 9.6 kBd.

The node LIN connectivity is implemented using the Freescale 1.3 driver software package (see Reference [11.]), which is available on the Freescale web pages (see Reference [10.]) free of charge.

In the project directory, the Freescale LIN 1.3 driver package is represented by folders as follows:

- Driver configuration: This folder includes the master.cfg and master.id files, which together enable the LIN driver configuration. Those files are user accessible to allow the Master node LIN connectivity modification,
- Driver source: This folder includes the LIN driver source files.

6.3.2 Application Software

The application LIN Master software enables control of the scheduling and contents of transmitted and received LIN frames via intelligent GUIs running in the FreeMASTER tool (see Section 2.2, “System Features”, Section 7.2, “GUIs General Description”, and Appendix A, “Messaging Strategy”).

The LIN Master software enables running the MM908E624 board LIN Slave application software in several different modes:

- Normal mode (see Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction” and Section 7.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Description”)
- Keep Window Speed mode (see Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction” and Section 7.2.2, “GUI “Window Lift Parameters Configuration” Description”)
- PWM Control mode (see Section 2.2.3, “GUI “The MM908E624 Board OUTs Controlled by the LIN Bus” Introduction” and Section 7.2.3, “GUI “The MM908E624 OUTs Controlled by the LIN Bus” Description”)

The LIN Master application software is represented in the project directory by the folder:

- Sources:
  - master.c and master.h (an application main file)
  - lin_control.c and lin_control.h (LIN connectivity related files)

6.3.3 Application Software Configuration Files

The MC9S12C32 LIN connectivity can be configured in the *lin_control.h* file. The definitions are as follows:

- CHECKSUM_OVER_THE_ID: By defining, this enables the LIN frames checksum over the frame identificator and data calculation, to keep the frame structure LIN 2.0 Specification compliant. However, it is also necessary to mention that the LIN Slave software is not LIN 2.0 complaint because the node configuration, which is mandatory for the LIN 2.0 devices, is not implemented (see Reference [13.]). The MC9S12C32 LINkit board can act as a LIN 2.0 compliant device, if the LIN 2.0 drivers are implemented in node (see Reference [14.] and Reference [15.]).
- SLAVE_WAKEUP_MASTER: The Freescale LIN 1.3 driver does not includes the procedure for the LIN Master wake-up, when the LIN Slave has issued the LIN Wake-Up frame. This define enables reading ESCI module received data. The routine for a LIN Master wake-up by a LIN Slave can be found in the *lin_control.c* file. This define has to be always defined to ensure proper LIN Master functionality.
- LIN_PHYS_LAYER_MC33399: Define this, if the MC9S12C32 LINkit board is populated by a MC33399 LIN physical layer[1].
- LIN_PHYS_LAYER_MC33661: Define this, if the MC9S12C32 LINkit board is populated by a MC33661 LIN physical layer[2].

1. If the LIN_PHYS_LAYER_MC33399 define is uncommented, the LIN_PHYS_LAYER_MC33661 define has to be commented.
2. If the LIN_PHYS_LAYER_MC33661 define is uncommented, the LIN_PHYS_LAYER_MC33399 define has to be commented.

6.4 LIN Master Software Description

6.4.1 Application Software Main Code

The LIN Master software enables control of the MM908E624 board LIN Slave by the LIN bus via intelligent GUIs running in the FreeMASTER tool (see Section 2.2, “System Features”). The LIN Master software flow chart is depicted in Figure 19.

The LIN Master issues standard communication LIN frames in a predefined time schedule, which equals 50ms between two following frame transmissions. If it is necessary, the time scheduling can be modified in the LIN Master software. The LIN frames identifier and data field contents depend on the required LIN Slave service (see Appendix A, “Messaging Strategy”). Otherwise, the LIN Master is also capable of configuring the LIN Slave Window Lift application strategic parameters via the LIN Master Request frame (0x3C) and uploading the parameter data back to the LIN Master via the LIN Slave Response frame (0x3D). Both actions are controlled via GUI (see Section 2.2.2, “GUI “Window Lift Parameters Configuration” Introduction” and Section 7.2.2, “GUI “Window Lift Parameters Configuration” Description”).

The LIN Master software enables control of the LIN bus communication data flow and the current LIN nodes status. It allows the user to Run and Stop LIN frames issue, Sleep and Wake-up the LIN bus.

The LIN bus can be woken-up either by the LIN Slave or by the LIN Master. When the LIN Slave has woken-up the LIN bus, the LIN Master node is switched from the LIN Sleep to LIN Run state and it starts issuing LIN frames. When the LIN bus frames issue is stopped, the status of the LIN bus becomes Idle. If the LIN bus state is Idle for more than 2.6 seconds[1], the LIN nodes should automatically enter the low power Sleep mode. The LIN Master software includes a LIN bus no activity counter to enable recognition, that the LIN nodes have entered Sleep mode. This provides the user with information on current LIN Slave nodes status.

To check whether the LIN nodes correctly respond to a LIN Master request, basic LIN error handling was added to the application software. The errors evaluation is based on reading the LIN error flags, that are periodically updated by LIN 1.3 driver software. The application software evaluates the transmit error, which reflects if either more than one node is currently transmitting or the LIN bus signal wire is connected to the power supply wire, and the receive error, which informs the user that the currently processed LIN node communication failed.

1. The LIN1.3 Protocol Specification (see Reference [6.]) assigns the LIN bus Idle timeout as 25000 of T_{bit} time. An application LIN communication runs at 9.6KBd, i.e. $T_{bit} = 1/ f_{BR} = 1 / 9600 = 104us$. The LIN bus timeout $T_{out} = 25000 * 104us = 2.6sec$.

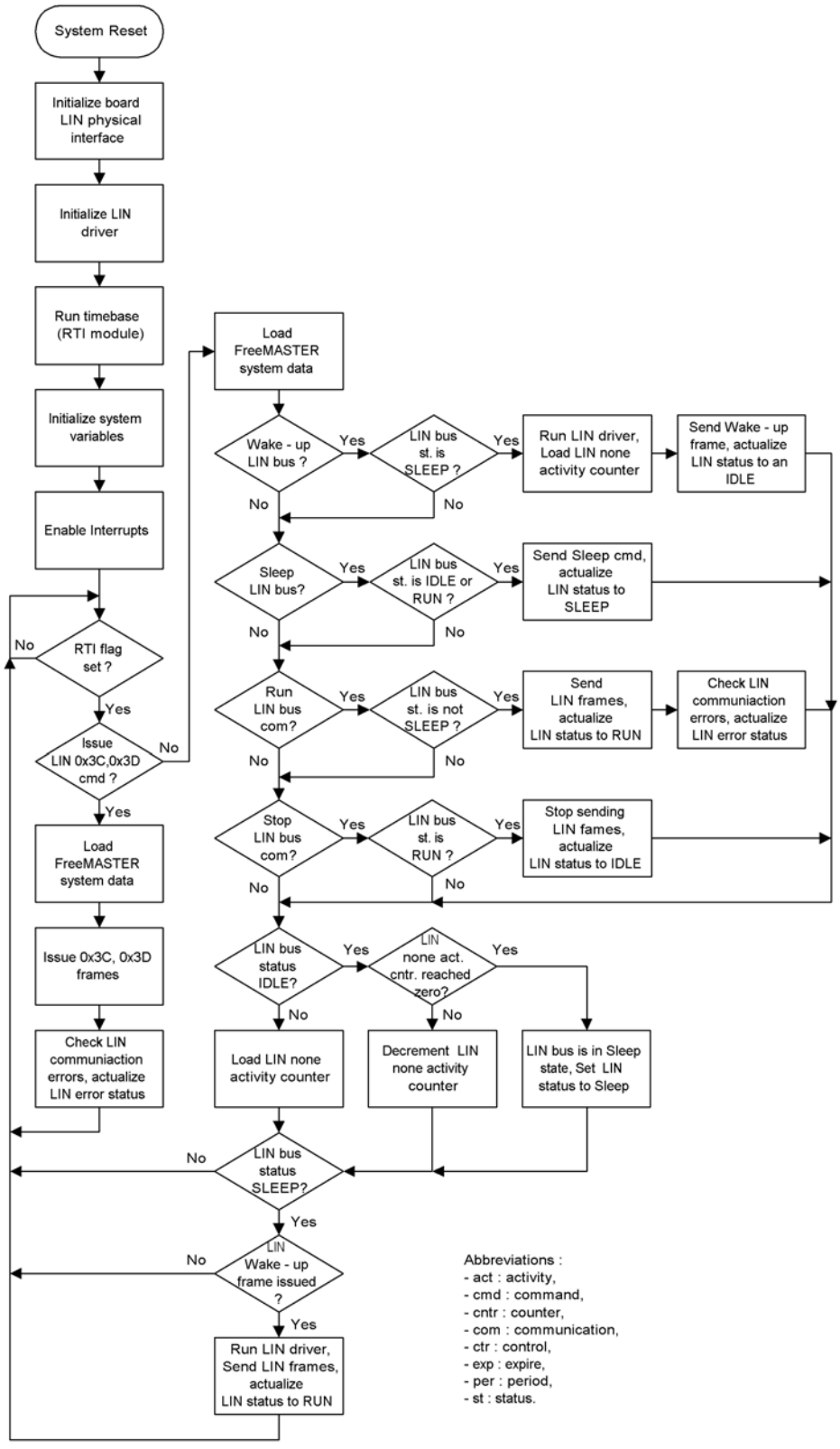


Figure 19. LIN Master Software Flow Chart

7 User Interface Description

7.1 Introduction

The Graphical User Interface (GUI) was created to allow the user easy application control. There are several GUIs to introduce the MM908E624 board as a Window Lift controller and a General Purpose Output PWM duty cycle and voltage polarity controller. The GUIs are described in the following sections:

- Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction”
- Section 2.2.2, “GUI “Window Lift Parameters Configuration” Introduction”
- Section 2.2.3, “GUI “The MM908E624 Board OUTs Controlled by the LIN Bus” Introduction”

GUIs run in the FreeMASTER tool (see Section 3.1, “FreeMASTER Tool”)

7.2 GUIs General Description

7.2.1 GUI “Window Lift Application Controlled by the LIN Bus” Description

This GUI (see Figure 20) offers the user to control the MM908E624 board as the Car Door Window Lift application.

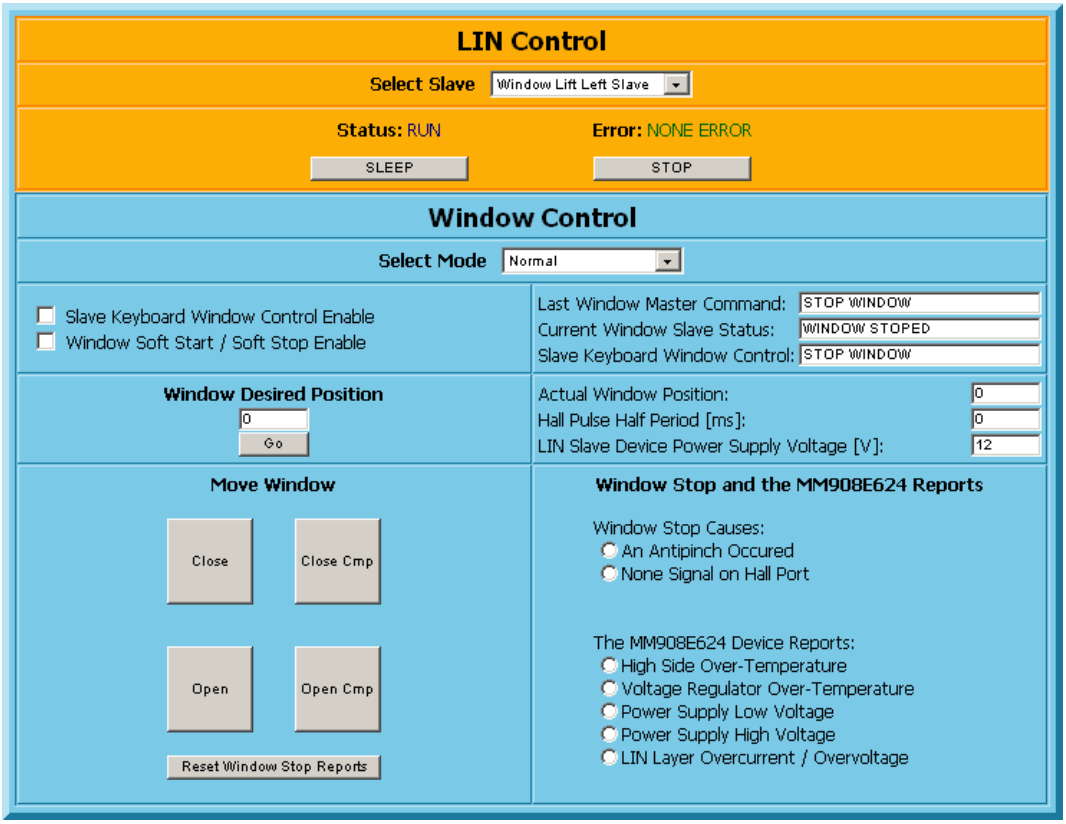


Figure 20. GUI “Window Lift Application Controlled by the LIN Bus”

The system features are as follows:

- LIN Control (GUI orange background):
  - **combo box “Select Slave”**: Roll down and select by clicking on the desired combo box item, which LIN Slave device has to be controlled (Right or Left Window Lift Slave).
  - **note “Status”**: It displays the current LIN bus Status (Run / Idle / Sleep).
  - **note “Error”**: It displays the current LIN bus communication errors (None Error / None Response -> LIN Slave not responding on issued frame header / Transmitter Issue -> the LIN bus signal wire is corrupted or more than one device is issuing LIN frames at the same time.).
  - **button “Run / Stop”**: Run or Stop LIN frames issue by a single click on the push button. The button displays the next possible change of the current LIN bus Status.
  - **button “Wake / Sleep”**: Wake-up or Sleep the LIN bus by a single click on the push button. The button displays the next possible change of the current LIN bus Status.
- Window Control (GUI blue background):
  - **combo box “Select Mode”**: Select “Normal” or “Keep Window Speed” MM908E624 board mode by rolling down and clicking on the desired combo box item. During the “Keep Window Speed” mode, the window glass movement speed is kept independent of the system power supply voltage variation (The Car Door Window Lift platform has to be adapted to run the Keep Window Speed mode, i.e. the Window Lift platform DC motor has to offer enough of a wide margin of PWM duty cycle control with an adequate window glass movement capability to run the window glass speed control.
  - **check box “Slave Keyboard Window Control Enable”**: Check the box, if window position control by the MM908E624 board push button keyboard (Open Window / Close Window / Stop Window)[1] is needed.
  - **check box “Window Soft Start / Soft Stop Enable”**: Check the box if the window Soft Start or Soft Stop feature is required (done by Car Door Window Lift DC motor PWM ramp control).
  - box “Window Desired Position”: Here, enter the Window Glass desired position and then click on the push button “Go”.
  - button “Go”: By clicking on the button, the window glass is going to move to the position specified in the box situated above the button.
  - **buttons “Close” and “Open”**: By clicking on the button, the window opens or closes. The button displays the next window action (Stop moving the window) initialized by the next single click.
  - **buttons “Close Cmp” and “Open Cmp”**: By clicking on the button, the window glass is going to close or open completely.
  - **button “Reset Window Stop Reports”**: If an “An Antipinch Occurred” or “None Signal on Hall Port” occurs, by clicking on the button, the MM908E624 board report is cleared and the board is prepared for the another command execution.

1. If the window is controlled by the MM908E624 on-board keyboard, then the GUI window control is disabled. To enable GUI window control, uncheck the checkbox “Slave Keyboard Window Control Enable“.

— **box “Last Window Master Command”**: This box displays the currently transmitted LIN Master command (Stop Window / Open Window / Close window / Open Window Completely / Close Window Completely / Window To Preset Position).

— **box “Current Window Slave Status”**: This box displays the current LIN Slave status (Window Stopped / Closing Window / Opening Window / Window Closed / Window Opened / Desired Window Position Reached).

— **box “Slave Keyboard Window Control”**: This box displays the MM908E624 board push button keyboard state. If the check box “Control Window by Slave keyboard” is checked, the keyboard controls the window position (Open Window / Close Window / Stop Window).

— **box “Actual Window Position”**: It displays the current window glass position.

— **box “Hall Pulse Half Period”**: It displays the current Hall pulse half period in milliseconds. This can be used for the MM908E624 board “Keep Window Speed” mode demonstration, because of the Hall pulse period independence of the power supply voltage variety.

— **box “LIN Slave Device Power Supply Voltage”**: It displays the current MM908E624 board power supply voltage in volts. The measurable supply voltage range is from 8 to 15V.

— dot box “An Antipinch Occurred”: It displays, whether a window glass antipinch occurred (the window glass reached an obstacle).

— dot box “None Signal on Hall Port“: It displays, whether the Hall signals are present on the MM908E624 board in the correct order.

— **The MM908E924 Device Reports**: It displays current MM908E624 analog die error reports.

7.2.2 GUI “Window Lift Parameters Configuration” Description

This GUI (see Figure 21) is closely linked to that previously described in Section 2.2.1, “GUI “Window Lift Application Controlled by the LIN Bus” Introduction”. It enables configuring the strategic parameters of the Car Door Window Lift application and so allows the user to easily rebuild an application for another Car Door Window Lift platform with similar features as the one used.

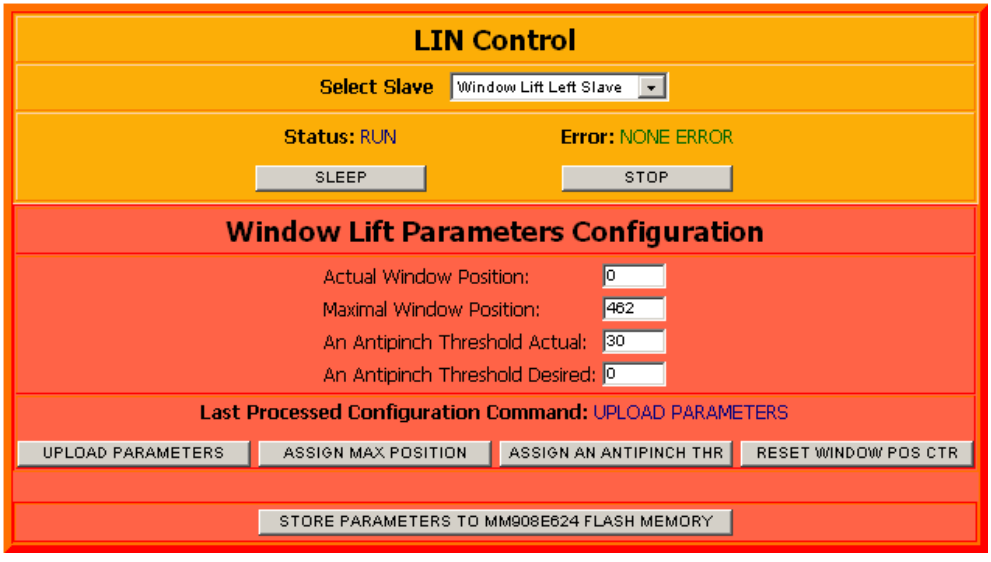


Figure 21. GUI “Window Lift Parameters Configuration”

The GUI offers the following features:

- LIN Control (GUI orange background):
  - **combo box “Select Slave”**: Select by rolling down and clicking on the desired combo box item, which LIN Slave device to control (Right or Left Window Lift Slave).
  - **note “Status”**: It displays the current LIN bus Status (Run / Idle / Sleep).
  - **note “Error”**: It displays the current LIN bus communication errors (None Error / None Response -> LIN Slave not responding on Issued frame header / Transmitter Issue -> the LIN bus signal wire is corrupted or more than one device is issuing LIN frames at the same time.).
  - **button “Run / Stop”**: Run or Stop LIN frames issue by a single click on the push button. The button displays the next possible change of the current LIN bus Status.
  - **button “Wake / Sleep”**: Wake-up or Sleep the LIN bus by a single click on the push button. The button displays the next possible change of the current LIN bus Status.
- Window Lift Parameters Configuration (GUI red background):
  - **button “Upload Parameters”**: By clicking on this button, the LIN Master issues the LIN Master Request command (ID 0x3C) request with the requirement of the Window Lift parameters structure upload of the LIN Slave node selected by “Select Slave” combo box. The LIN Slave sends the data on the Slave Response frame (ID 0x3D) header issue.
  - **button “Assign Max Position”**: The Window Lift parameter structure maximal position is loaded with the actual window glass position counter value (done by 0x3C command). Then the parameter structure is uploaded to update the LIN Master with new parameter structure data (done by 0x3D command). This is helpful, if necessary to assign a new window maximal position (position of closed window)/
  - **button “Assign An Antipinch Thr”**: The Window Lift antipinch threshold assignment. The assigned value is defined in box “An Antipinch Threshold Desired” (done by 0x3C command). After the 0x3C command is processed, the parameter structure is uploaded to update the LIN Master with new structure data (done by 0x3D command)/
  - **button “Reset Window Pos Ctr”**: Clicking on the button causes the window glass position counter reset (done by 0x3C command). Then the LIN Master is updated with new Window Lift parameter structure data(done by 0x3D command)/
  - **button “Store Parameters to MM908E624 Flash Memory”**: Clicking on this button causes storage of the Window Lift parameter structure to MM908E624 MCU Flash memory to prevent structure data loss.

7.2.3 GUI “The MM908E624 OUTs Controlled by the LIN Bus” Description

This GUI (see Figure 22) introduces the MM908E624 board as a general purpose output voltage polarity and PWM duty cycle controller.

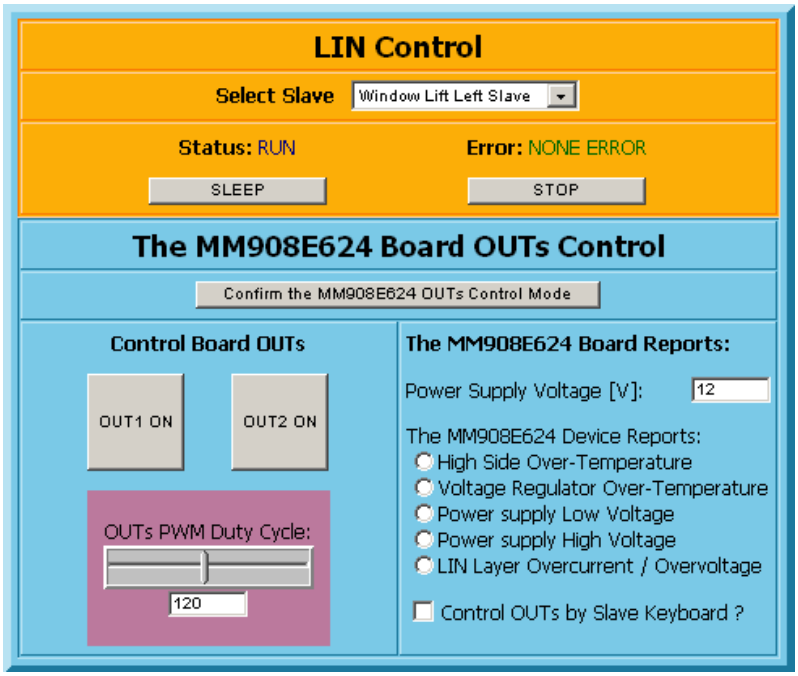


Figure 22. GUI “The MM908E624 Board OUTs Controlled by the LIN Bus”

Offered features are as follows:

- LIN Control (GUI orange background):
  - **combo box “Select Slave”**: Select by rolling down and clicking on the desired combo box item, which LIN Slave device to control (Right or Left Window Lift Slave).
  - **note “Status”**: It displays current LIN bus Status (Run / Idle / Sleep).
  - **note “Error”**: It displays current LIN bus communication errors (None Error / None Response -> LIN Slave not responding on Issued frame header / Transmitter Issue -> the LIN bus signal wire is corrupted, or more than one device is issuing LIN frames at the same time.)
  - **button “Run / Stop”**: Run or Stop LIN frames issue by a single click on the push button. The button displays the next possible change of the current LIN bus Status.
  - **button “Wake / Sleep”**: Wake-up or Sleep LIN bus by a single click on the push button. The button displays the next possible change of the current LIN bus Status.

- The MM908E624 Board OUTs Control (GUI blue background):
  - button “Confirm the MM908E624 OUTs Control Mode”: By clicking on this button, the MM908E624 board is configured to be a general purpose OUTs controller.
  - button “OUT1 ON” and “OUT2 ON”: By these buttons, the MM908E624 board OUTs state is controlled. The buttons display the next possible change of the OUTs state.
  - **OUTs PWM Duty Cycle**: By the slide bar control or the box number entry (0 - 255), it is possible to change the OUTs PWM duty cycle. The board initial OUTs PWM duty equals 120, and the step equals 1 (duty cycle step approximately equals 0.4%).
  - **box “Power Supply Voltage”**: It displays current MM908E624 board power supply voltage in volts. The measurable supply voltage range is from 8 to 15V.
  - **The MM908E624 Device Reports**: It displays the current MM908E624 analog die error reports.
  - **check box “Control OUTs by Slave keyboard ?”**: Check the box if it is needed to control the MM908E624 board OUTs state by on-board keyboard[1].

1. If MM908E624 board OUTs are controlled by the MM908E624 on-board keyboard, then the GUI OUTs control is disabled. To enable GUI OUTs control, uncheck the checkbox “Control OUTs by Slave keyboard ?“.

8 Conclusion

This AN covers the MM908E624 implementation in the Window Lift / Sun Roof application. The primary goal of this application is to show the performance and capabilities of the MM908E624 single package solution. The other goal is to introduce the MC9S12C32 MCU as a LIN Master device and demonstrate the LIN 1.3 connectivity, that in this case is implemented in the node using the Freescale LIN 1.3 drivers (see Reference [11.]).

The complete application software for the LIN Window Lift Master (MC9S12C32) and LIN Window Lift Slave (MM908E624) is downloadable as an AN3147SW package.

The MM908E624 MCU (908EY16) memory consumption is calculated in Table 1.

Table 1. Particular MM908E624 Code Sizes

MCU Memory Type	MCU Memory Size	LIN1.3 Software Occupies	Application Software Occupies	Totally Occupied Area	Free Space	Free Space
Flash	15872 bytes	1012 bytes	3290 bytes	4302 bytes	11570bytes	73%
RAM	512 bytes	57 bytes	68 bytes	125 bytes	387 bytes	75%

The application uses these MM908E624 MCU (908EY16) peripherals:

- 1-of-8 A/D converter channels
- TIMA and TIMB timers
- TBM module
- IGC module
- ESCI module
- SPI module

The rest of the MM908E624 stuff is free and can be utilized by the user for additional application purposes (see MM908E624 block diagram on Figure 4).

9 References

1. MM908E624: *MM908E624 Integrated Triple High-Side Switch with Embedded MCU and LIN Serial Communication for Relay Drivers*, Revision 5.0, 01/2005, Freescale Semiconductor datasheet
2. MC68HC908EY16/D: *MC68HC908EY16 Advance Information*, Revision 7.0, 5/2004, Freescale Semiconductor datasheet
3. MC9S12C32: Product > Microcontrollers > 16bit > HCS12 > MC9S12C32 16bit Microcontroller product summary page, Freescale Semiconductor Web Page, http://freescale.com
4. MC33399: *MC33399 Automotive LIN Physical Interface*, Revision 4.0, 2/2005, Freescale Semiconductor datasheet
5. MC33661: *MC33661 Local Area Network (LIN) Enhanced Physical Interface with Selectable Slew Rate*, Revision 4.0, 2/2005, Freescale Semiconductor datasheet
6. LIN Specification Package, Revision 1.3, 12th December 2002, LIN consortium, http://www.lin-subbus.org/
7. FreeMASTER Product Summary Page: Product > Digital Signal Processors & Controllers > DSP Development Tools > Software Tools > FreeMASTER, Freescale Semiconductor Web Page, http://freescale.com
8. AN2573/D: *LINkits Evaluation Boards*, Revision 1.0, 11/2003, Freescale Semiconductor datasheet
9. IDC Product Summary Page: Product > Analog > Connectivity Solution > Embedded MCU + Power, Freescale Semiconductor Web Page, http://freescale.com
10. General Freescale LIN web page: , http://freescale.com/LIN, Freescale Semiconductor Web Page
11. Freescale LIN 1.3 Driver page:, http://freescale.com/LIN, Design Tools > LIN1.x Design Tools > Software > Device Drivers, Freescale Semiconductor Web Page
12. EB195/B: *How to Configure the Reset Pin on the MC68HC11*, Freescale Semiconductor Engineering Bulletin, 1999
13. LIN Specification Package, Revision 2.0, 23rd September 2003, LIN consortium, http://www.lin-subbus.org/
14. AN2767: *LIN 2.0 connectivity on Freescale 8/16bit MCUs using Volcano LTP*, Freescale Semiconductor datasheet
15. LIN 2.0 Driver page: , http://freescale.com/LIN, Design Tools > LIN2.0 Design Tools, Freescale Semiconductor Web Page
16. MSE908EY16_1L38H: *MC68HC908EY16 Mask Set Errata for Mask 1L38H, Revision 1.0*, 5/2005, Freescale Semiconductor Errata

10 Acronyms

ADC	Analog to Digital Converter module
AN	Application Note
API	Application Program Interface
CAN	Controller Area Network
COP	Computer Operating Properly module
CW	CodeWarrior
eLIN	Enhanced LIN
ESCI	Enhanced Serial Communication Interface module
EVB	Evaluation Board
GND	Power Supply Ground Terminal
IC	Integrated Circuit
ICG	Internal Clock Generator module
IDC	Intelligent Distributed Control
I/O	Input / Output ports
LED	Light Emitting Diode
LIN	Local Interconnect Network
LINkit	EVB for LIN development
Master	The LIN Master node controls the LIN connectivity
MCU	MicroController Unit
MW	MetroWerks
OUTs	MM908E624 board relay outputs OUT1, OUT2
PC	Personal Computer
SIO	LIN Serial Input Output Signal Wire
Slave	The LIN Master controls the LIN Slave node task
TBM	TimeBase Module
VSUP	Power Supply Voltage

Appendix A
Messaging Strategy

Node Name for Signal Provider	ID [0..5]	LIN ID Field (w/parity)	Frame Name	Frame Description	Frame Size Bytes)	Sig #	Signal Name	Signal Description	Signal Length (bits)	Signal Start Bit	Signal Initial Value	Raw Value Range
<published_by>	<frame_id> (0 to 63)	–	<frame_ name>	–	<frame_ size>		<signal_ name>	–	<signal_ size>	<signal_ offset>	<init_ value>	–
WL_MASTER	0x0A 0x1A)	0xCA (0x1A)	WL_L_CMD (WL_R_CMD)	Window Lift Left (Right) Command	4	0	wCmd	Window Command	4	0	0	0 – WINDOW STOP 1 – WINDOW CLOSE 2 – WINDOW OPEN 5 – WINDOW CLOSE COMPLETELY 6 – WINDOW OPEN COMPLETELY 8 – WINDOW GO TO DESIRED POSITION
						1	outPwmCtrEnb	OUTs PWM Duty Cycle Control	1	4	0	0 – PWM CONTROL DISABLED 1 – PWM CONTROL ENABLED (During Keep Window Speed and PWM Control mode, PWM_CTR bit has to be set, otherwise the MM908E624 board is not working properly.)
						2	boardMode	MM908E624 Board Mode	2	5	0	1 – NORMAL MODE 2 – KEEP WINDOW SPEED MODE (Set PWM_CTR bit also) 3 – PWM CONTROL MODE (Set PWM_CTR bit also)
						3	pwmDes	MM908E624 Board OUTs Desired PWM Duty Cycle	8	8	0	MM908E624 board OUTs PWM CUTY CYCLE (0–255) (LIN Slave evaluates this signal during PWM Control mode only)
						4	windowPosDes	Window Desired Position	16	16	0	WINDOW POSITION DESIRED (0–65535) (LIN Slave evaluates this signal during Normal and Keep Window Speed modes)
WL_LEFT_SLAVE (WL_RIGHT_SLAVE)	0x0B (0x1B)	0x8B (0x5B)	WL_L_STAT (WL_R_STAT)	Window Lift Left (Right) Status	3	0	wStat	Current Window Status	4	0	0	0 – WINDOW STOPPED 1 – WINDOW CLOSING 2 – WINDOW OPENING 5 – WINDOW CLOSED COMPLETELY 6 – WINDOW OPENED COMPLETELY 8 – WINDOW REACHED DESIRED POSITION
						1	wAntipinch	Window Antipinch	1	4	0	0 – NOT OCCURRED 1 – OCCURRED If an antipinch occurred, LIN Master has to send Window Stop Command to clear LIN Slave antipinch flag.
						2	wSReq	LIN Slave Keyboard Window Action Request	2	5	0	1 – WINDOW CLOSE 2 – WINDOW OPEN 3 – WINDOW STOP
						3	acE624Stat	MM908E624 Analog Die Status	5	8	0	MM908E624 ANALOG DIE REGISTER D2–D6 BITS COPY IN BIT ORDER FROM LSB TO MSB AS FOLLOWS: – HSST (High-Side over-temperature) – VDDT (Voltage Regulator over-temperature) – LVF (Power supply low voltage) – HVF (Power supply high voltage) – LINFAIL (LIN layer overcurrent / overvoltage)
						4	hallErr	Hall Port Error	1	13	0	0 – NOT OCCURRED 1 – OCCURRED If a Hall error occurred, LIN Master has to send Window Stop Command to clear LIN Slave Hall error flag.
						5	powerSupVolt	MM908E624 Board Power Supply Voltage	8	16	0	powerSupVolt (0–255) * 0.1 = MM908E624 board power supply voltage [V}

Node Name for Signal Provider	ID [0..5]	LIN ID Field (w/parity)	Frame Name	Frame Description	Frame Size Bytes)	Sig #	Signal Name	Signal Description	Signal Length (bits)	Signal Start Bit	Signal Initial Value	Raw Value Range
<published_by>	<frame_id> (0 to 63)	–	<frame_ name>	–	<frame_ size>		<signal_ name>	–	<signal_ size>	<signal_ offset>	<init_ value>	–
WL_LEFT_SLAVE (WL_RIGHT_SLAVE)	0x0C (0x1C)	0x4C (0x9C)	WL_L_INFO (WL_R_INFO)	Window Lift Left (Right) Info)	3	0	windowPosAct	Window Position Actual	16	0	0	ACTUAL WINDOW POSITION (0–65535)
						1	hallPulseWidth	Hall Pulse Signal Period	8	16	0	hallPulseWidth (0–255) * 0.1 = hall pulse period [ms]
WL_MASTER	0x3C	0x3C	MasterReq	LIN Master Request Command	8	0	slaveCfgCmd	Requred LIN Slave Action	8	0	0	0x00 – SLEEP COMMAND 0x80 – ASSIGN MAXIMAL POSITION (Copy Window Actual Position counter value to Window Maximal Position variable) 0x81 – ASSIGN NEW WINDOW ANTIPINCH THRESHOLD (Tick Difference Threshold value) 0x82 – RESET WINDOW ACTUAL POSITION COUNTER 0x83 – STORE PARAMEERS TO LIN SLAVE FLASH MEMORY 0x84 – DOWNLOAD PARAMETERS FROM LIN SLAVE TO LIN MASTER
						2	linSelectSlave	Select LIN Slave	8	8	0	0 – WINDOW LIFT LEFT LIN SLAVE 1 – WINDOW LIFT RIGHT LIN SLAVE
						3	tickDifThrDes	New value of Tick Difference Threshold	16	16	0	NEW WINDOW ANTIPINCH THRESHOLD (0–65535)
WL-LEFT_SLAVE (WL_RIGHT_SLAVE)	0x3D	0x7D	SlaveResp	LIN Slave Response Command	8	0	slaveCfgCmdLast	Last Required LIN Slave Action	8	0	0	0x00 – SLEEP COMMAND 0x80 – ASSIGN MAXIMAL POSITION (Copy Window Actual Position counter value to Window Maximal Position variable) 0x81 – ASSIGN NEW WINDOW ANTIPINCH THRESHOLD (Tick Difference Threshold value) 0x82 – RESET WINDOW ACTUAL POSITION COUNTER 0x83 – STORE PARAMEERS TO LIN SLAVE FLASH MEMORY 0x84 – DOWNLOAD PARAMETERS FROM LIN SLAVE TO LIN MASTER
						1	tickDifThrAct	Tick Difference Threshold	16	8	0	CURRENT WINDOW ANTIPINCH THRESHOLD(0–65535)
						2	posMax	Window Maximal Position	16	24	0	MAXIMAL WINDOW POSITION (Position, where the Window is closed completely) (0–65535)
						3	windowPosAct	Window Actual Position	16	40	0	ACTUAL WINDOW POSITION (0–65535)
						4	captureEdge	Last Capturred Hall Sensor Edge	2	56	0	MM908E624 board Hall port (pin 2) Hall Sensor 1 signal LAST CAPTURED EDGE: 1 – RISING 2 – FALLING

Appendix B
System Setup

B.1 General Overview

The system can be run in two different system setups. The first system setup introduces an application as the MM908E624 board OUTs high current controller. This setup is recommended for running the system as the Window Lift application, because of the Window Lift platform high current requirement (typically up to 15A). The second system set up shows an MM908E624 board OUTs low current control (typical up to 300mA). It's suitable for the MM908E624 OUTs voltage polarity and PWM duty cycle control demonstration.

NOTE

Section B.2, "System Hardware Setup" introduces several possible hardware configurations. Please, follow one of those system setups and avoid connecting a power supply source to both boards (908E624 board and MC9S12C32 LINkit) at once. It can cause damage of used stuff.

B.2 System Hardware Setup

B.2.1 “High Current” System Setup

The system setup for the high current OUTs control is depicted in Figure 23. The system is supplied from the MM908E624 board J1 connector to enable running the high current board loads as the Window Lift and Run Roof. The system parameters:

- Nominal power supply voltage: 12V (range from 8 to 18V)
- Power supply current capability: It depends on the controlled load parameters. For the MM908E624 board Window Lift control it is recommended to use a power supply with up to 20A current sourcing capability.

NOTE

It is necessary to keep a correct power supply voltage polarity, otherwise the MM908E624 condenser C1 (reservoir of energy for the Window Lift DC motor currents peak management) explodes (see schematic in Figure 8) and the 908E624 board, including the MC9S12C32 LINkit, can also be damaged.

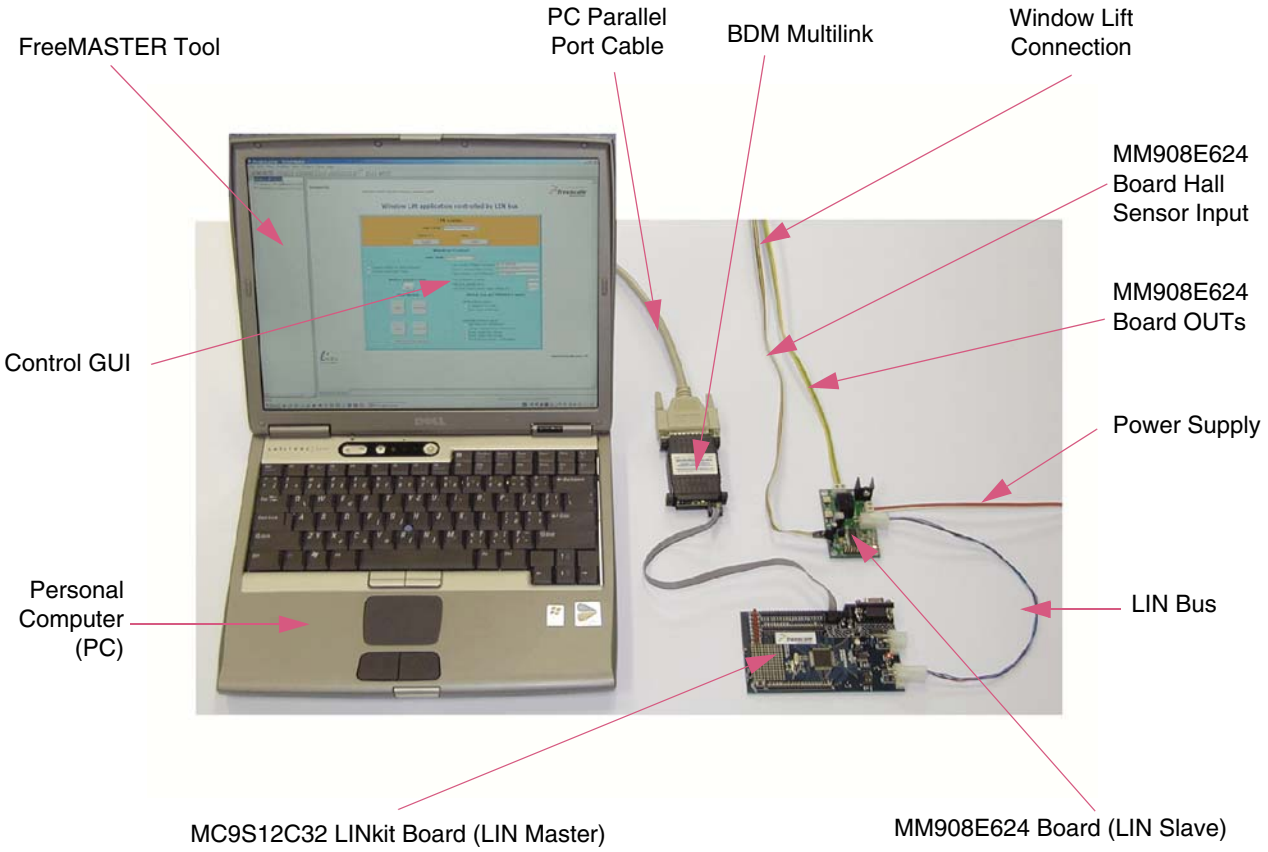


Figure 23. “High Current” System Setup

B.2.2 “Low Current” System Setup

To enable running the system without any “huge” power supply source, the “Low Current” system setup was created (depicted in Figure 24). The system is supplied from an external source via MC9S12C32 LINkit power supply connector PW1 (see schematic in Figure 11). This system setup mostly corresponds to the reality, where the LIN nodes are supplied from the LIN Master device. However, because of the MC9S12C32 LINkit board low current power supply capability, limited by board 500mA F1 fuse, the maximal MM908E624 board OUTs current must not exceed a maximal limit of 300mA. The remaining current capability of 200mA is resolved for the MC9S12C32 LINkit board stuff and also the BDM multilink device power supply. The nominal power supply voltage equals 12V, however, it can float in range from 8V to 18V.

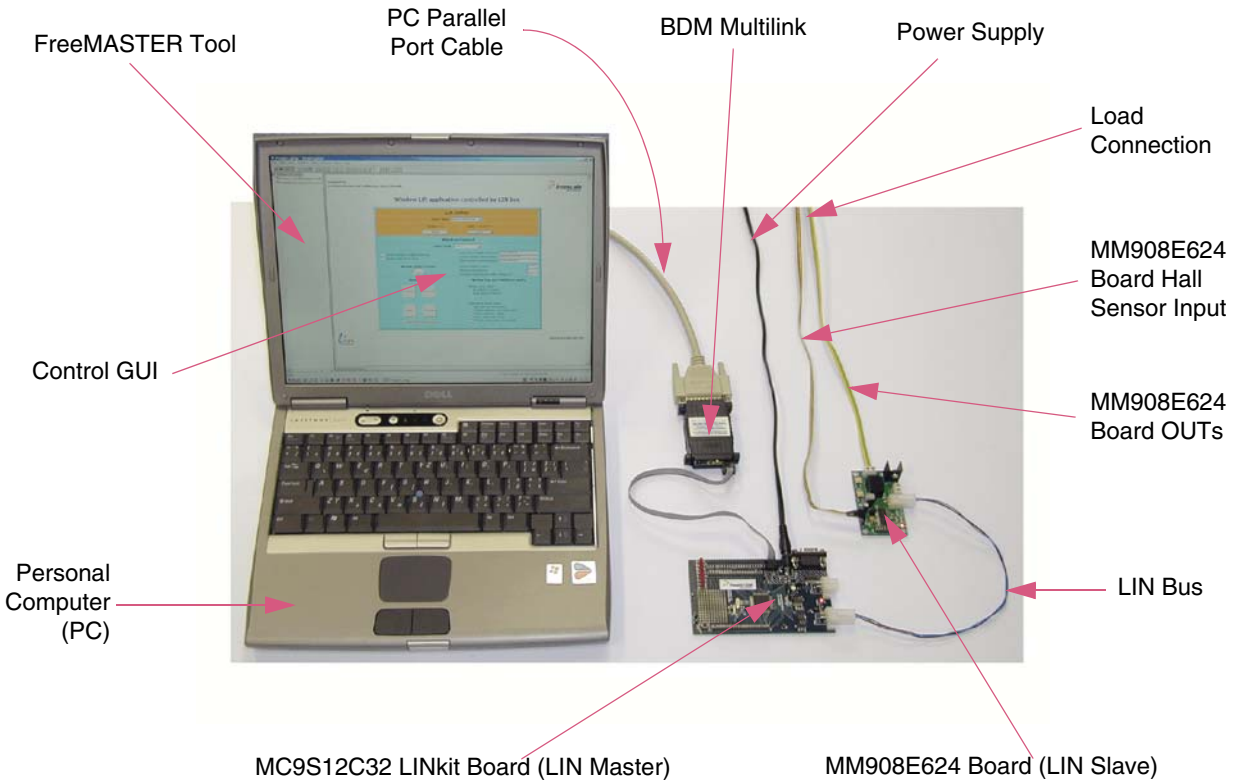


Figure 24. “Low Current” System Setup

B.2.3 MM908E624 Board Jumper Setting

The MM908E624 board includes jumper JP1, which controls the MM908E624 analog die Window Watchdog (see Section 4.1, “MM908E624 Integrated Triple High-Side Switch with Embedded MCU and LIN Serial Communication for Relay Drivers”). If the JP1 board jumper is opened, the analog die Window Watchdog is enabled.

The MM908E624 application software does not includes the routine for the MM908E624 analog die Window Watchdog service, therefore as Figure 7 shows, the JP1 has to be always closed.

B.2.4 MM908E624 Board Window Lift Platform Connection

The MM908E624 board connectors interface is depicted in Figure 9. It is necessary to connect the Hall sensors to the MM908E624 board in the correct order, depending on the required window glass action, as Figure 25 and Figure 26 show. Notice, that the Hall Sensor signal period is not only dependent on the actual window glass position, but also on the window glass movement direction. If the window glass is opening, the Hall Sensor signal pulse period is lower than during the window close execution. This is caused by gravitation, as the window glass is attracted to the earth.

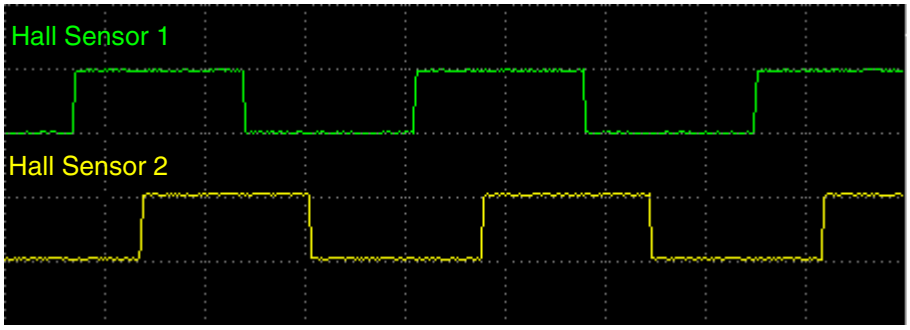


Figure 25. Hall Port Signals Order During Window Closing

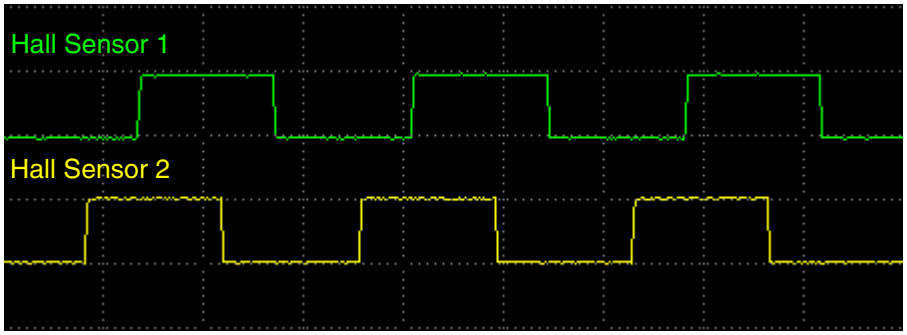


Figure 26. Hall Port Signals Order During Window Opening

B.3 System Software Setup

B.3.1 MM908E624 Board Software

The MM908E624 software is written mainly in C language, except some routines which are written in HC08 assembler. The MM908E624 board code is generated using Metrowerks CodeWarrior for HC(S)08 devices compiler, version 3.1. The code is downloaded to the MM908E624 device by the board J3 standard 16-pin MON08 multilink connector header (see MM908E624 board description in Figure 7).

B.3.2 MC9S12C32 LINkit Board Software

The MC9S12C32 software is mainly written in C language, except some routines which are written in HC12 assembler. The MC9S12C32 board code is generated using Metrowerks CodeWarrior for HC(S)12 devices compiler, version 3.1. The code is downloaded to the MC9S12C32 device by the LINkit board H1 standard 6-pin BDM multilink connector header (see MM908E624 board description in Figure 10).

B.3.3 FreeMASTER Tool

The FreeMASTER tool has to be installed on a PC together with the BDM plug-in, which creates the data line between the PC and MC9S12C32 LINkit board. The FreeMASTER tool can be downloaded from the Freescale FreeMASTER web pages Reference [7.]. To obtain the BDM plug-in, see advanced information on the Freescale FreeMASTER web pages.

B.4 Application Executing

When the application hardware and software setup is complete, connect the power supply voltage and run the FreeMASTER tool by double clicking on the Project.pmp file. If the FreeMASTER is running, go to the tool bar (see Figure 27), select Project -> Options and check if the “Comm” and “MAP Files” folders setup is correct (see Figure 28 and Figure 29). If the folder setup is not correct, correct it and then click on “Save Project” to save the new project setting. For a proper application run, the “Stop Communication” button should not be pushed, as Figure 27 shows.

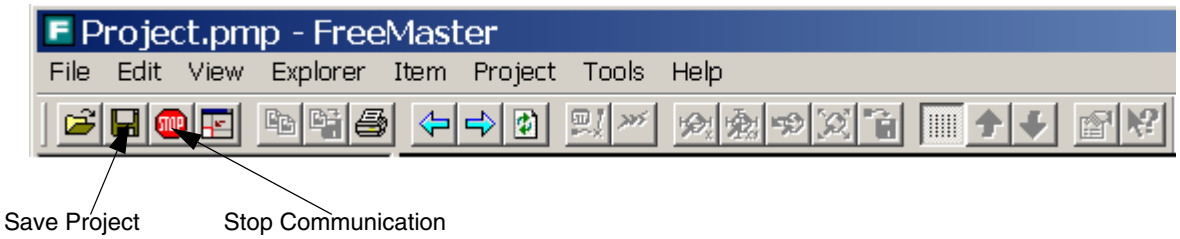


Figure 27. FreeMASTER Tool Bar

If the communication between the FreeMASTER and MC9S12C32 is not running yet, try to disconnect and then again connect the BDM multilink connector to the LINkit board. Then close the FreeMASTER tool and re-open it.

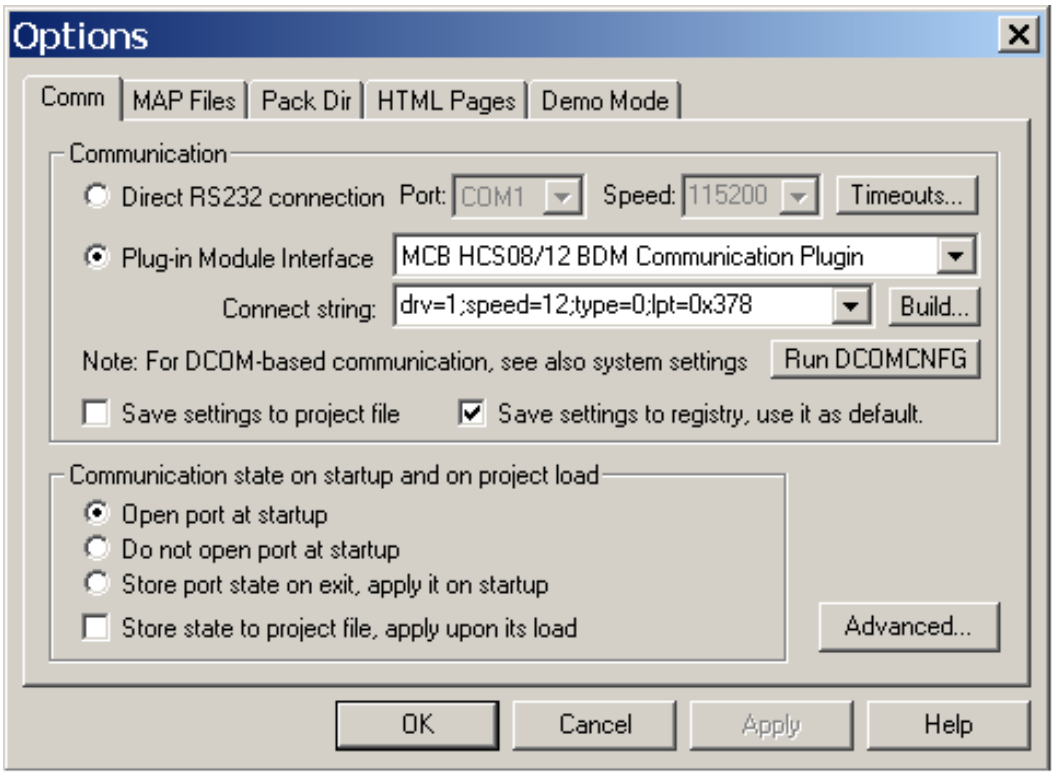


Figure 28. FreeMASTER "Comm" Folder Setup

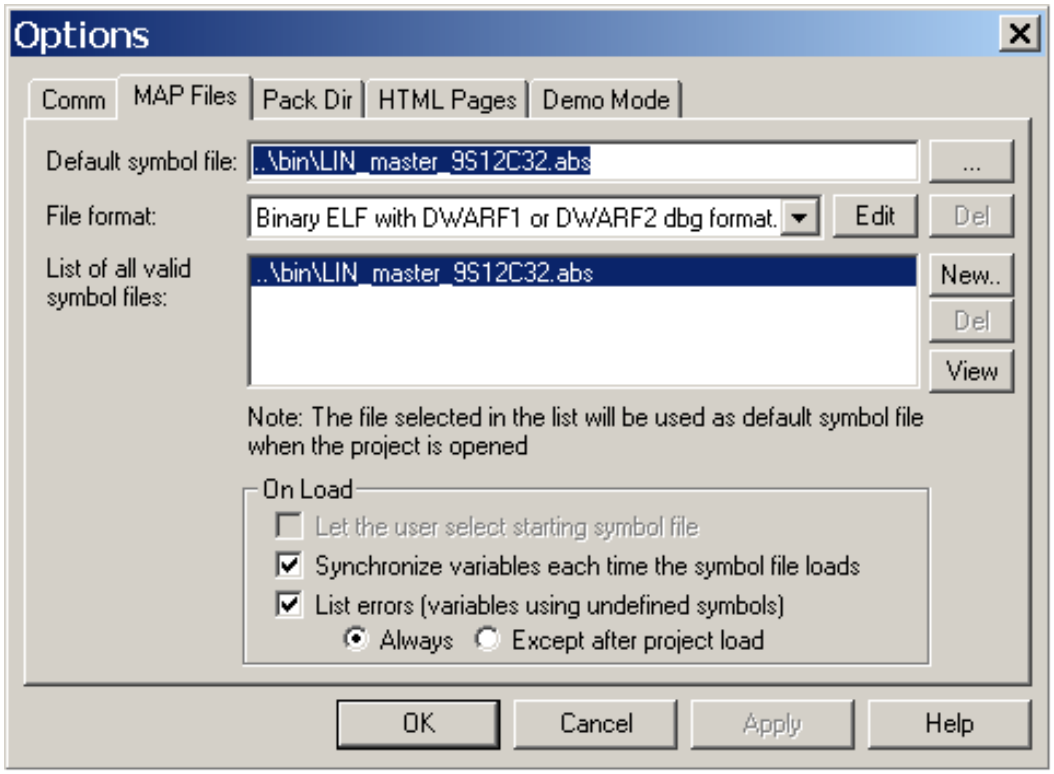


Figure 29. . FreeMASTER "MAP Files" Folder Setup

If the MM908E624 boards OUTs current is too high and the transition behavior of the power supply source is so slow, the drop in the power supply source voltage may cause the MC9S12C32 LINkit reset. To avoid this, connect between the MM908E624 board and MC9S12C32 LINkit board LIN power supply inputs diode and increase the LINkit board decoupling capacitor C1 (see schematic in Figure 11) value up to 220uF/35V (Figure 30).

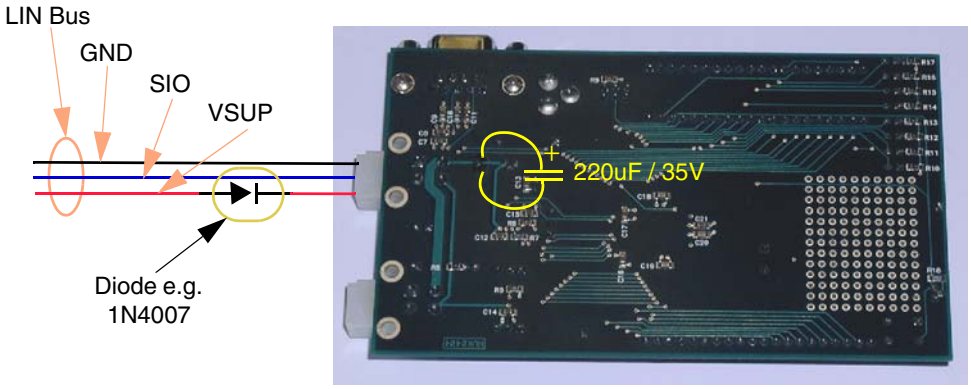


Figure 30. MC9S12C32 LINkit Board Bottom View

If it is necessary to measure the MM908E624 supply current, e.g. in device Sleep mode, disconnect the diode D1 (see schematic in Figure 8) from the MM908E624 board and use these free pads for the current measurement device connection[1] (see Figure 31).

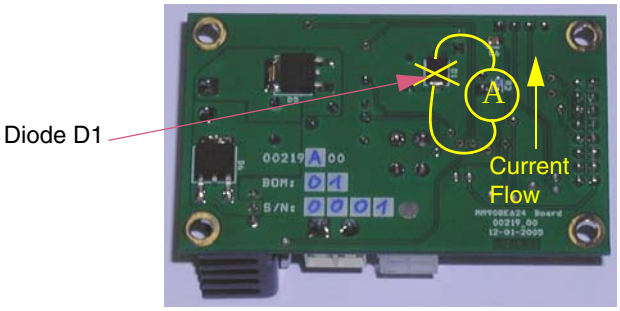


Figure 31. MM908E624 Board Bottom View

1. Be aware, that the MM908E624 is not protected against incorrect power supply voltage polarity now. Don't push the MM908E624 board keyboard (S1, S2), otherwise the measured power supply current value will not be correct.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

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

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

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

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

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

AN3147
Rev. 0.2, 11/2005

freescale™ semiconductor