Freescale Semiconductor

Order this document by
AN1688/D

AN1688

MISC BUS SLAVE SWITCH NODE

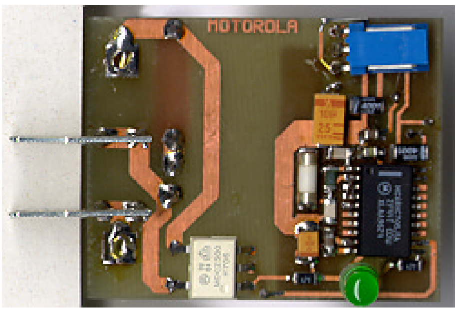
By Jaromir Chocholac
Roznov Systems Application Laboratory
MOTOROLA - Czech Republic

1 INTRODUCTION

This Slave switch node, based on the very low cost MC68HC705J1A MCU, is used to illustrate the implementation of a software SCI within a communication protocol, as described in AN1667/D. It is an example of how to drive the mini solid state relay MOCZ500 from an MCU. It can be a good solution for the electro-mechanical relays replacement by the solid state relays. There are no mechanical movements, in solid state relays, than no contact bounce. An important advantage of solid state relays is the unlimited number of operations.

The Slave switch node, described in this application note, is capable of driving loads up to 500 mA, on line voltages from 20V to 280V AC (rms) by its mini solid state relay with zero voltage activation and can be used in the applications such as pumps, valves and lighting.

Low-cost MCUs often do not include a hardware SCI. Using these MCUs is interesting in Niche Area Networks. To perform asynchronous serial communication, software emulation of the SCI must be used. What is remarkable about the system described here is that a complete data link protocol stack is implemented and executed, concurrent with a real application, on a sub-$1 MCU.



2 SWITCH NODE HARDWARE DESCRIPTION

A key feature of this node is the driving capability for a AC Load up to 0.5 Amp based on data sent from the master node through the bus. The switching capability is done by the mini solid state relay MOCZ500. It is a reliable, low cost solid state alternative to low current AC electromechanical relays. This mini solid state relay can be driven from most microcontroller outputs, because of 10 mA LED trigger current.

There are four parts in this Slave switch node: Load driver; Microcontroller; Bus transceiver; Power supply. Figure 2-1 shows the Slave switch node block diagram.

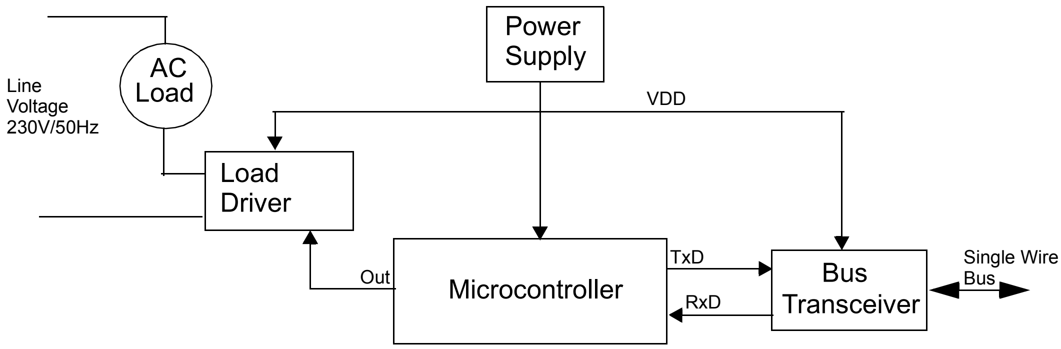


Figure 2-1. Slave Switch Node Block Diagram

Figure 2-2 shows the Slave switch node schematic. The node is supplied through connector J1 from the external 12V DC power supply. A load is connected to the Line voltage 230V/50Hz in this example, but the Line voltage can be from 20 to 280 V rms. The J1 connector pin description is in Table 2-1.

Table 2-1. J1 connector pin description

PIN NUMBER	PIN NAME	DESCRIPTION
1	BUS	BUS LINE
2	GND	GROUND
3	POWER	POWER SUPLY 12V

The MC78L05 voltage regulator (U3) on the node provides 5 volts to the circuitry. The MC68HC05J1A MCU (U1) drives the Slave switch node. The MOCZ500 mini solid state relay (U4) is used as a load driver. The LED (D3) is controlled by the MISC communication software to indicate the condition of the slave node. The LED will flash slowly, when the communication between the operating Slave switch node and the master operates correctly. When a sync message is received the LED changes its state. (it is about once every second). If there is a problem with the data link the LED will flash faster.

The MC33188 bus interface circuit was used for this Slave switch node. The PB4 (TxD) and PB5 (RxD) pins of the MCU are connected to the bus interface. The IRQ pin is connected to the PB5 pin to ensure the interrupt when the data are coming. It is necessary for SW_SCI implementation. The MC33188 is an interface circuit designed for automotive applications. It is capable of interfacing to several bus types. There is K_bus setting on the node as a MISC physical layer. The bus line is internally short circuit protected. The circuit has an automatic stand-by mode, but it must be disabled for the MISC protocol.

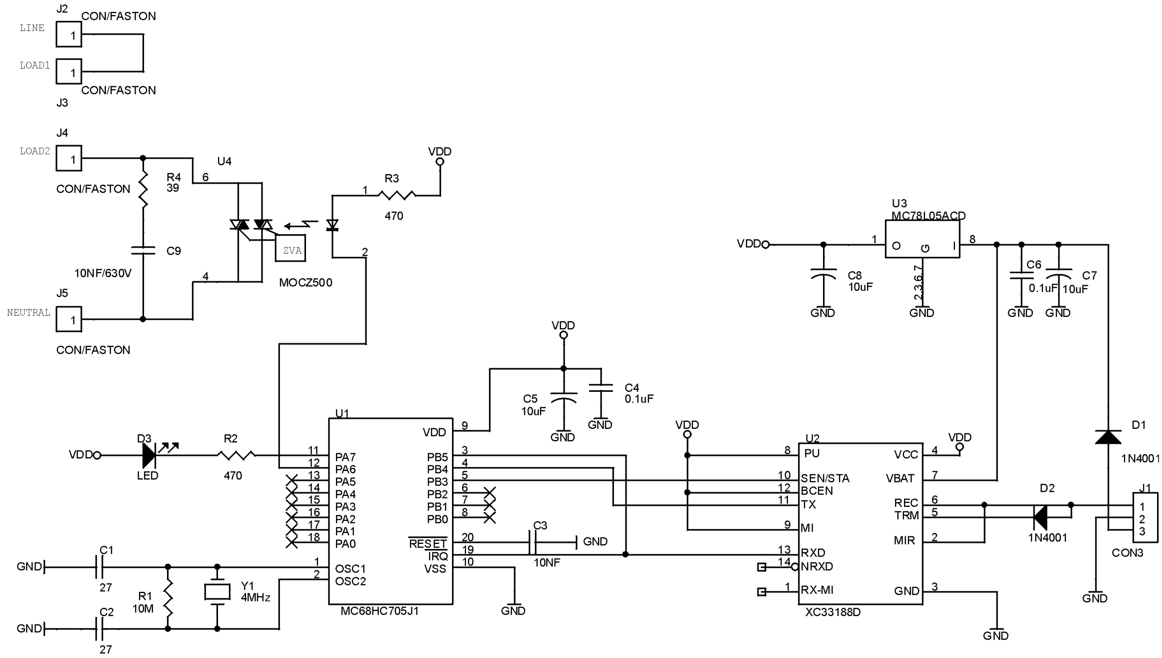


Figure 2-2. MISC MC68HC705J1A Switch Node Schematic

Table 2-2. Switch Node Bill of Materials

Device Type	Quantity	Value	Reference
Ceramic Capacitor	2	27pF	C1,C2
Ceramic Capacitor	2	0.1uF	C4,C6
Ceramic Capacitor	1	10nF	C3
Electrolytic Capacitor	3	10uF	C5,C7,C8
Capacitor	1	10nF/630V	C9
Diode	2	1N4001	D2,D1
LED	1	Green	D3
Connector	1	G88/34	J1
Connector	2	FASTON	J2,J3,J4,J5
Mini Solid State Relay	1	MOCZ500	U4
Resistor 0.25W	1	10M	R1
Resistor 0.25W	1	470	R2,R3
Resistor 0.5W	1	39	R4
IC	1	MC68HC705J1A	U1
IC	1	MC33188	U2
IC	1	MC78L05ACD	U3
Ceramic Resonator	1	4MHz	Y1

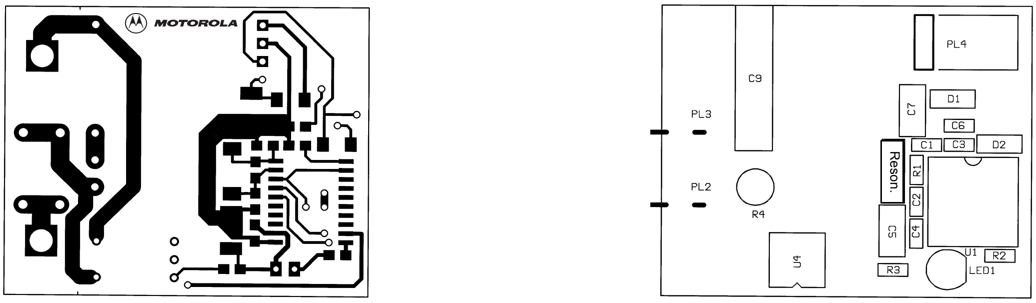


Figure 2-3. Slave Switch Node - Component Side

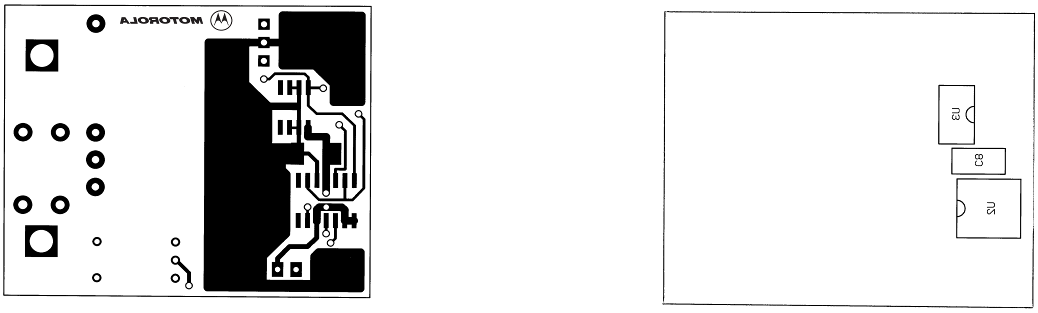


Figure 2-4. Slave Switch Node - Solder Side

3 SLAVE SWITCH NODE CONTROLLING

The Slave switch node is controlled from the master by commands. The command is part of the push header frame of a message. Figure 3-1 shows the MISC header frame format. The header frame consists of two fields - address and command. The push address field contains the message destination.

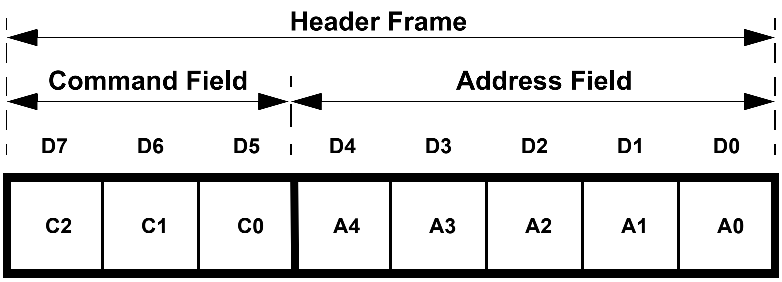


Figure 3-1. MISC Header Frame Format

Each message is standardized and comprises two fields. The Master sends a Push field and then the selected slave responds by a Pull field. The composition of a MISC message is shown in Figure 3-2.

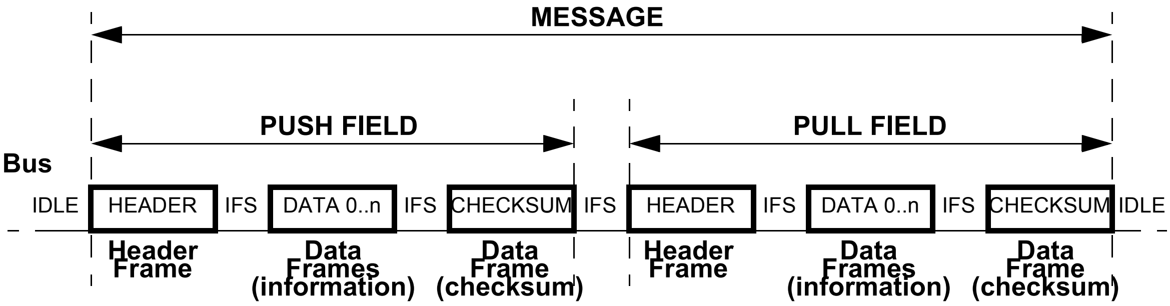


Figure 3-2. MISC message composition

For this application a message consists of six bytes: Header frame; one data frame; checksum of push field; header frame; one data frame; checksum of pull field. With this size of message, it is possible to have eight commands using the three bits of the Command field. We are using only two commands (Load ON (001), Load OFF (010) in this application.

Each slave node in the network, must have its own unique address. The commands described below, as an example, have implemented address 6. All Push and Pull fields contain three bytes. Their meanings follow.

Command 000 - No action

Field	Header byte	Data byte	Checksum byte
Push	$06	$00	done by the Data Link
Pull	$06	$00	done by the Data Link

Command 001 - Load ON

Field	Header byte	Data byte	Checksum byte
Push	$26	$00	done by the Data Link
Pull	$26	$00	done by the Data Link

Command 010 - Load OFF

Field	Header byte	Data byte	Checksum byte
Push	$46	$00	done by the Data Link
Pull	$46	$00	done by the Data Link

Command 011 - No action

Field	Header byte	Data byte	Checksum byte
Push	$66	$00	done by the Data Link
Pull	$66	$00	done by the Data Link

Command 100 - No action

Field	Header byte	Data byte	Checksum byte
Push	$86	$00	done by the Data Link
Pull	$86	$00	done by the Data Link

Command 101 - No action

Field	Header byte	Data byte	Checksum byte
Push	$A6	$00	done by the Data Link
Pull	$A6	$00	done by the Data Link

Command 110 - No action

Field	Header byte	Data byte	Checksum byte
Push	$C6	$00	done by the Data Link
Pull	$C6	$00	done by the Data Link

Command 111 - No action

Field	Header byte	Data byte	Checksum byte
Push	$E6	$00	done by the Data Link
Pull	$E6	$00	done by the Data Link

4 SLAVE SWITCH NODE SOFTWARE DESCRIPTION

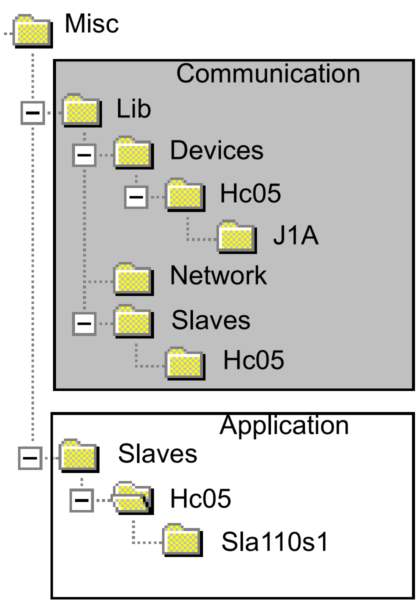


Figure 4-1. Directory Structure

The software can be divided into two levels - the Application and Communication levels. Figure 4-1 shows the recommended directory structure.

Communication level software is obtained from the library and object files are linked to the application files. The MAKE.BAT is a batch file for user friendly creation of the Motorola S-record format file. Only the application level software is described in this document the communication level software is described in AN1667/D.

4.1 MISC\LIB DIRECTORY

This directory contains all of the library files that are referenced by MISC application programs. The information within these files is generally common across many application programs so placing the information in one place avoids duplication and aids maintenance. The library files are split into directories according to the device type and application attributes.

4.2 MISC\SLAVES DIRECTORY

This directory contains all of the files which are specific to MISC Slaves. The files are split into directories according to the device type and application attributes.

4.3 SOURCE CODE

The example software source files (SWITCH.ZIP) you can find on the http://design-net.sps.mot.com. The source code is written in MASM 4.9 but it is easy to port to assemblers delivered by 3rd parties.

4.3.1 Application Files Brief Description

After the software installation, you can find these files in Sla110s1 directory.

MAIN.ASM	- Main application example assembler source code
MAIN.H	- Main application example assembler header file
MAININT.ASM	- Main application interrupt service routines assembler source code
MAINRAM.ASM	- Main application RAM storage assembler source code
MAINSTUB.ASM	- Main application Data Link stub routines assembler source code
MAINSTUB.H	- Main application Data Link stub header file
MAKE.BAT	- Batch file to make or link any or all of the files in this directory

5 INSTALLING SOFTWARE

There is no need to run real installation. Just the SWITCH.ZIP file unzip operation is needed. Once the file is unzipped using "pkunzip.exe -d switch.zip", the file structure as you can see in Figure 4-1 will be created. The user may use the Slaves.s19 file to burn in the MC68HC705J1A. If the user makes some changes in the application files, he can use make.bat file, to generate new Slaves.s19 file.

6 CONCLUSION

Open systems based on a distributed architecture have features for future. Multiplex wiring protocols are good solutions for this purpose. Low cost micro-controllers will find many applications in the Industrial and White Goods field. In this area we can have tiny networks - "Niche Area Networks (NAN)". Frequently these networks are totally self-contained within a single machine. The use of a low cost micro-controller is one of the best ways to implement a low cost distributed architecture at the sensor/actuator level. There will be many other opportunities to use this solution in the future when Home and Building automation will emerge.

REFERENCES

AN1667/D - Software SCI implementation to the MISC communication protocol

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

freescale™ semiconductor

AN1688/D