SGS-THOMSON MICROELECTRONICS

APPLICATION NOTE

POWER LINE MODEM APPLICATION REMOTE CONTROL USING ST7537 AND ST6

By Joël HULOUX, Patrice MOREL

SUMMARY **Page**

I	**INTERFACE BETWEEN ST7537 AND ST626X**	3
I.I	INTERFACE	3
I.2	PINS DESCRIPTIONS	3
I.2.1	RSTO and NMI	3
I.2.2	Carrier Detect	3
I.2.3	Rx/Tx.	3
I.2.4	Rxd and Txd	3
I.2.5	Watchdog	3
I.2.6	Oscin	3
I.2.7	Other ST6265 pins	3
I.2.7.1	RESET Pin	3
I.2.7.2	PA4..pA7 Pins	3
I.3	SOFTWARE INITIALISATION	4
I.4	TIMER 1 FOR 1200BPS	4
I.5	INTERRUPT ROUTINES	5
I.5.1	NMI interrupt	5
I.5.2	Receive Interrupt	5
I.5.3	Timer 1 Interrupt	5
II	**TRANSMISSION METHOD**	5
II.1	MAIN ASPECT OF COMMUNICATION	5
II.1.1	Asynchronous 1200bps.	5
II.1.2	Carrier Detect	5
II.1.3	Programs Specifications	5
II 2	PACKED MESSAGE.	6
II.3	C.S.M.A. PROTOCOL SPECIFICATIONS	6
II.4	REAL TIME AND COMMUNICATION.	7
II.5	FRAME DEFINITION.	7
II.5.1	Standard Frame	7
II.5.2	Acknowledge Frame	7
II.6	COMMUNICATION REGISTERS DEFINITION	8
II.6.1	Communication Registers	8
II.6.2	Network/Application Register	8
II.6.3	Parity Register.	9
II.7	MAINS FLOW	9
II.7.1	Main Flow	9
II.7.2	Reception Flow	10
II.7.3	Acknowledge Control Flow	10

III	**APPLICATION : DIMMER CONTROL**	10
III.1	GENERAL DESCRIPTION	10
III.2	SCHEMATICS	11
III.3	PINS DESCRIPTION & SOFTWARE INITIALISATION	13
III.3.1	Slave Board	13
III.3.2	Master Board	13
III.4	DIMMER CONTROL	13
III.4.1	General Description	13
III.4.2	Triac Specification	14
III.4.3	Timing Specification	14
III.5	APPLICATION SOFTWARE	14
III.5.1	Display Application	14
III.5.2	Slave System	14
III.5.2.1	AR Timer Configuration	14
III.5.2.2	Registers for Power Control	15
III.5.2.3	Dimmer Flow	15
III.5.2.4	Push-button	15
III.5.2.5	Potentiometers	15
IV	**NETWORK**	16
IV.1	NETWORK SPECIFICATION	16
IV.1.1	Communication between 2 Units	16
IV.1.2	Communication with Several Units	16
IV.1.3	Timinig Control	17
IV.2	EXAMPLE OF IMPLEMENTATION OF SOFT CARRIER DETECTOR	17
IV.3	IMPLEMENTATION OF C.S.M.A. ON ST6265	18
V	**CONCLUSION**	19
VI	**ANNEXE A : ST6x PROGRAMS**	19
VII	**ANNEXE B : SCHEMATICS**	20

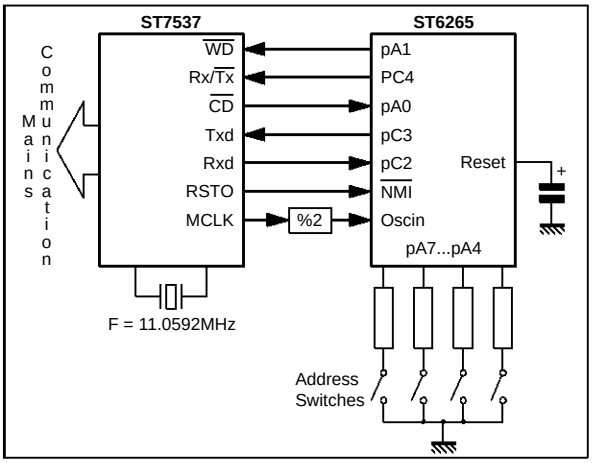
PRELIMINARY

All benefits and features of the ST7537CFN have been detailed in Application Note AN655. I suggest the reader to read this one before starting with this new application.

I- INTERFACE BETWEEN ST7537 AND ST626x

SGS-THOMSON is now introducing with this interface between the power line modem ST7537 and the low cost family ST6MCU a wide range of applications for home automation. This module allows communication between distant equipment by the mains and then remote control can be done in an easy way. This include applications like lighting dimmer, heater control, or phone remote system.

Figure 1 : Hardware Connection Between ST7537 and ST6265



I.1 - Interface

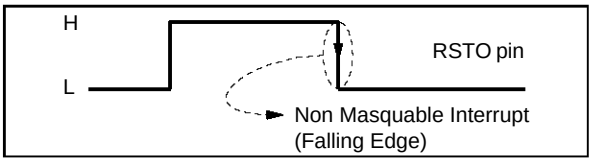
The choice of these ST6 pins is not the only one, but the software suggested is written for this interface.

I.2 - Pins Descriptions

I.2.1 - RSTO and NMI

The RSTO is the reset output of the ST7537. It set to high when the supply voltage is lower than a limit (typically 7.6V) or when no negative transition occurs on the watchdog input for more than 1.5s. Then, the RSTO is going back to low level. This falling edge is used to make an NMI on the ST6 chip. The NMI vector (number 0) must be the same as the reset vector in order to comply with Home System specifications.

Figure 2 : RSTO Generate a NMI



I.2.2 - Carrier Detect

The Carrier detect is driven low when the signal amplitude on the receive analog input is greater than a carrier detection level (typically 5mV). It has to be connected to an interrupt pin in order to start the receive program, even if the ST6 is doing something else. So the pin pA0 has to be programmed as an interrupt (with pull-up) to allow receiving.

I.2.3 - Rx/$\overline{Tx}$

The Rx/$\overline{Tx}$ pin is used to switch between receive and transmit mode. It has to be connected to an output port of the ST6, to allow the ST6 to switch between Rx and $\overline{Tx}$. pC4 has to be programmed as output port.

I.2.4 - Rxd and Txd

Digital datas are going over these pins. They have to be connected to pC2 and pC3 for the data transfer. Then, the programmer can use the PSI (Programmable Serial Interface) or work on the data values by programming pC3 as output and pC2 as input (without interrupt and with pull-up).

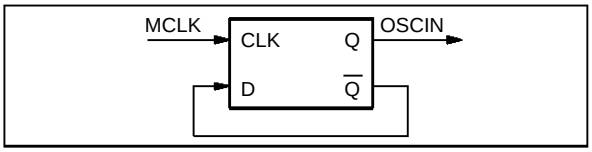
I.2.5 - Watchdog

This pin is connected to pA1, which must have a falling edge at least every 1.5s (MAX value). This feature has been included in ST7537 for security reason. So pA1 has to be set and reset at least every 1.5s (see RSTO pin description).

I.2.6 - Oscin

The maximum operating frequency of ST6265 microcontroller is 8MHz with a 5V DC supply (and 8.5MHz with 6V). In order to use the 11.0592MHz provided by the P.L.M., we must decrease this frequency. An easy way is to divide by 2 the master clock :

Figure 3 : The Flip-flop D-type Divide the Clock



A flip-flop D-type is used to divide MCLK by 2. So, the microcontroller has an input frequency of 5.5296MHz (quartz frequency divided by 2).

I.2.7 - Other ST6265 Pins

I.2.7.1 - RESET Pin

To provide a good initialisation, a 2.2µF capacitor is connected to the ST6265 reset pin. An internal 300kΩ is loading the capacitor during the power on. This provides a delay that allows power supply stabilisation. The high value of the resistor requires a tantalum capacitor type.

I.2.7.2 - pA4..pA7 Pins

These pins are used to enter an objet address (network configuration). pA4..pA7 are programmed in input with pull-up. Address number can be modified if necessary.

I.3 - Software Initialization

At reset state, all the ports are in input with pull-up, and interrupt register (IOR) is cleared, so the reset routine must configure the ports to comply with hardware connections.

Table 1 : Port Configuration

Register	Bits Values	Byte Value (e.g.)	Description
IOR	GEN (D4) set LES (D6) set	50 h	Enable all interrupts Level sensitive mode on interrupt #1 (port A)
DDRA	D0 res D1 set D4..D7 res	02 h	A0 in input A1 in output A4..A7 in input
ORA	DO set D1 set D4..D7 res	03 h	A0 interrupt with pull-up A1 in output push-pull set to 1 A4..A7 in input with pull-up
DRA	D0 res D1 set D4..D7 res	02h	
DDRB	D0..D7 set	FF h	B0..B7 in output
ORB	D0..D7 set	FF h	B0..B7 in push-pull output set to 1
DRB	D0..D7 set	FF h	
DDRC	D2 res D3 set C4 set	0C h	C2 in input C3 in output C4 in output
ORC	D2 res D3 set D4 set	0C h	C2 interrupt with pull-up C3 in output push-pull set to 1 C4 in output push-pull set to 1
DRC	D2 res D3 set D4 set	0C h	

The hexadecimal values are given only for example, these values will change due to the others pins programming (application software). In the program, every write to port A and port C must be done by writing a copy register, because these ports are in input and output, and this includes single bit operation (see ST6265 datasheet for further details).

I.4 - Timer 1 for 1200Bps

The MCU frequency is 5.5296MHz. With an internal clock divider set to 1, a prescaler set to 4 and a counter set to 96, the transmit and receive rate is 1200.08Bps (Figure 4).

The quartz is a 11.0592MHz with a accuracy of 100ppm. That means a final variation of 0.12Bps around 1200.08Bps.

When the timer 1 is not used for the baudrate this one is used for C.S.M.A. management (see II.3).

Table 2 : Baudrate Variation

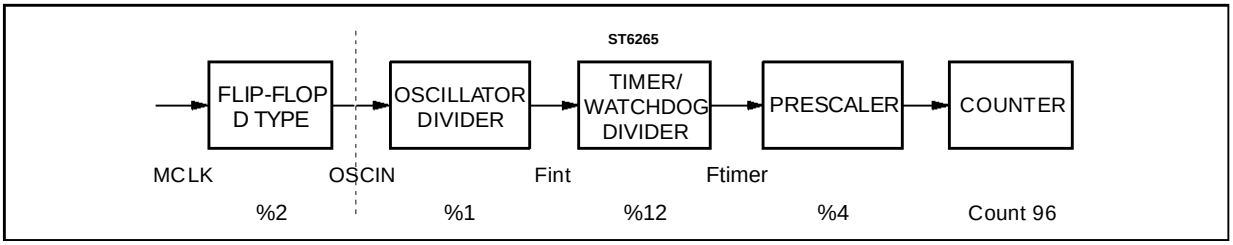
	Min.	Nom.	Max.	Unit
Rate	1,199.96	1,200.08	1,200.2	Bps

That means the following registers values :

Table 3 : Timer 1 and Oscillator Configuration

Register	Bit	Value
Oscillator control register OSCR	RS0 RS1	0 0
Timer status control register TSCR1	PS0 PS1 PS2	0 1 0
Timer counter register TCR1	D0 - D7	96 (decimal value)

Figure 4 : Dividing MCU Clock to Obtain 1200Bps



I.5 - Interrupt Routines

Interrupt vectors are mostly defined by the transmission program. NMI (vector #0) and the port A interrupt (vector #1) are used by the communication program. The port C (vector #2) does not need interrupt. The Timer 1 and ADC interrupt is used for communication timing (see CSMA recommendation).

AutoReload timer interrupt is also available.

Table 4 : Interrupt Programming

Vector #0	NMI	JP RESET
Vector #1	Port A and B	JP RECEIVE (or cascaded interrupt)
Vector #2	Port C and SPI	NOP and RETI
Vector #3	AR Timer	Available interrupt for application
Vector #4	ADC and Timer 1	Communication timing

I.5.1 - NMI Interrupt

The RSTO pin of the ST7537 is connected to the NMI pin of the ST6265 pin. The NMI interrupt provides the reset of the system.

I.5.2 - Receive Interrupt

This interrupt manages frame reception, that means :

- Power line modem pins control,
- Bit reception (synchronisation),
- Byte control,
- Frame control.

The interrupt program set bits in network/application register (NA-CTRL) then the main program is able to know the communication status by reading this register.

I.5.3 - Timer 1 Interrupt

This interrupt is enabled at the end of a frame reception or transmission to allow a delay between communication events (see C.S.M.A. specifications). It also leave time for application program.

II - TRANSMISSION METHOD

II.1 - Main Aspect of Communication

- Asynchronous 1200Bps,
- Transmit mode must be set for less than 1s (Cenelec specifications),
- Carrier detect is set between 4 and 6ms after the beginning of the Reception (the first 3 or 4 bits are lost) and is reseted in the same delay (an interference byte is received at the end of the frame). This delay corresponds to the demodulator trainning.

II.1.1 - Asynchronous 1200Bps

In order to transmit at 1200Bps, the transmit program needs a timer (TIMER 1) to send a bit every 1/1200s. In the receiving routine, the program is synchronised on every startbit, and use a timer (Timer 1) to read a bit every 1/1200s. This allows synchronisation on each byte, and the reception is aborted if carrier disappears between byte reception. During byte reception Txd is set to "0" in order to be independent of CD level (noise independant).

II.1.2 - Carrier Detect

The Figure 5 shows the timing diagram of a communication.

In order to allow a good transmission, the frame must contain a header that is more than 3 bits long and that contains no 0 (start bit) : FF h for example. An easy way to avoid the last bad bit is to know the frame length (fixed length for instance).

II.1.3 - Programs Specifications

In transmit mode, the Rx/$\overline{Tx}$ pin must be low, and set to high after the transmission of the frame. Transmission time must be shorter than 1s : that allows a maximum frame length of :

$$\frac{1200}{11} = 109 \text{ bytes (1 byte = 1 start bit + 8 data bits + 1 parity bit + 1 stopbit = 11 bits).}$$

Figure 5 : Flow Diagram of Digital Datas

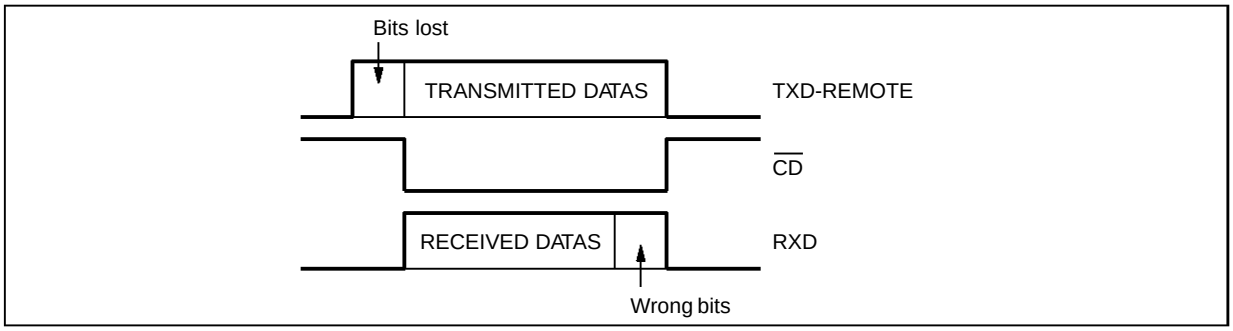


Figure 6 : Byte Format

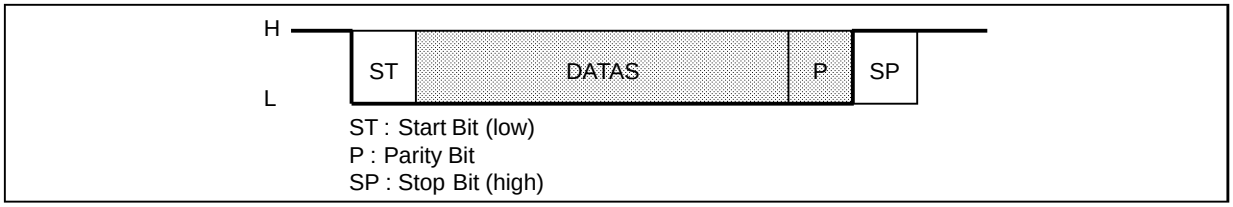
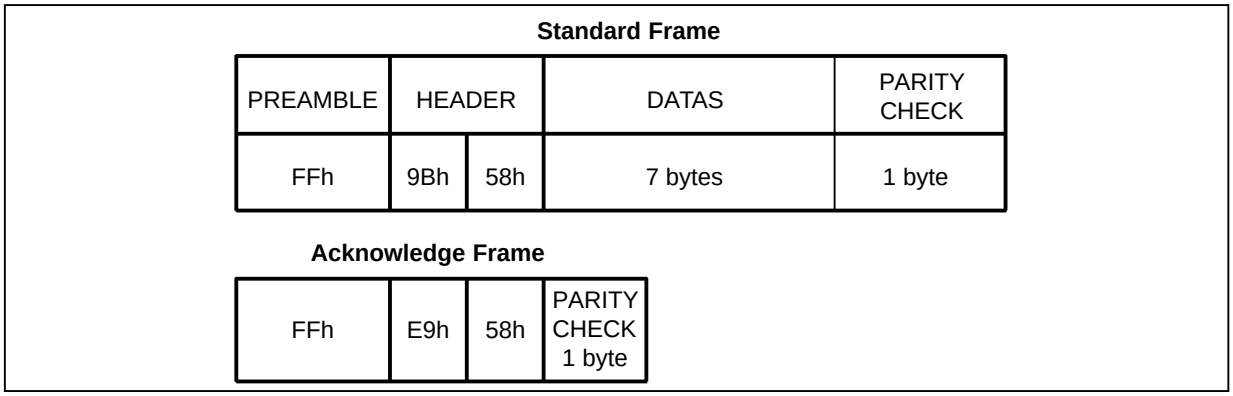


Figure 7 : Frames Format



II.2 - Packed message

The power line medium and the modulation employed require a special two level encapsulation mechanism :

- The byte level,
- The packet level.

The byte level control is a low level control, it only checks the start bit, the stop bit and the parity bit. The parity control can be done with byte shift and bit test (sla, jrr, jrs, ...) (Figure 6).

The packet level control is checking the received values. Preamble is used to synchronise and avoid loose of meaning datas. Header works out the frame type (Figure 7). For instance :

- 9B58h : Standard frame
- E958h : Acknowledge frame

The parity check byte is composed by all the parity bits of the data block of the frame (see parity register).

II.3 - C.S.M.A. Protocol Specifications

The protocol used is Carrier Sense Multiple Access Protocol. The telegram acknowledge is used to detect non-delivery. Before transmitting, the device must verify that there is no carrier on the network thanks to /CD, then send its message (standard frame) and wait for an acknowledge of the remote device (acknowledge frame). The device is able to check the transmission integrity by comparing the transmitted and the received parity check byte. The remote device is sending an acknowledge only if the packet is good (byte and packet control).

The C.S.M.A. protocol definition includes a timing for each stage of the communication. These values are given in the Table 5.

This is the theoretical definition of CSMA specifications. The programs proposed are using a register that is decreased in the main loop, providing a pseudo-random value (1 to 5). For the timing, the timer 1 counts for a delay (depending on the random register).

Table 5 : C.S.M.A. Specifications

Stage	Min. Time	Max. Time	Number of Values
Total duration of transmission		2s	
Duration of transmission after starting		1s	
Length to wait from the end of a remote transmission to initiate a transmission	85ms	115ms	7
Acknowledge sent after	0ms	30ms	Defined by software
Retry transmitting (after ack. time)	0ms	42ms	Uniformly distributed
Duration between two transmissions of the same device	125ms		

II.4 - Real Time and Communication

Time is one of the most important factor in this communication application :
- Mains is acting as a network (see C.S.M.A. specification),
- Home automation system requires a time delay below 0.5s.

So, communication time as to be taken into account. The table below gives values for message delivery (message + acknowledge) with the following values :
- Standard frame time : 100ms,
- Acknowledge frame time : 40ms,
- C.S.M.A. specifications.

In order to approximate to the reality, we have considered 1 error communication and/or a time waited to dispose of communication channel.

In Table 6, Tother is the time from the moment the object wants to transmit (but somebody is already transmitting) to the moment the communication channel is free.

A message needs about 155ms to be transmitted with acknowledge. In the worth case, even if 7 objects want to transmit in the same delay, all the messages will be delivered in less than 2s.

II.5 - Frame Definition

Here is the definition of frames implemented in ST6265 microcontroller.

II.5.1 - Standard Frame

Figure 8 : Standard Frame Format

PREAMBLE	HEADER	DATAS	PARITY CHECK

- Preamble : FFh
- Header : 9B58h
- Datas : Explained below
- Parity check : Parity bits of the data block

Figure 9 : Data Block Format

H.A.	C.B.	S.A.	D.A.	ORDER	DATA
2 bytes	1 byte	1 byte	1 byte	1 byte	1 byte

The Datas block is the real message. It contains the house address, to prevent from sending order to other houses objects. The source and destination addresses are used to define who's talking to who. The Control byte contains network facilities as priority, frame counter, ...
- H.A : (Home address) address of the house
- C.B : (Control byte) contain frame counter, priority, group command bit
- S.A. : (Source Address) address from the transmitter
- D.A. : (Destination Address) address of the receiver
- Order : Object of the message
- Data : Data byte eventually

II.5.2 - Acknowledge Frame

Figure 10 : Acknowledge Frame Format

PREAMBLE	HEADER	RECEIVED PARITY

- Header : E958h

The standard frame is 11 bytes long, that means 100ms at 1200Bps (with 11 bits by byte).

The acknowledge frame is 4 bytes long, that means 40ms at 1200Bps (with 11 bits by byte).

Table 6 : Message Delivery Time

Communication Time (ms)	No Communication Errors			One Communication Error		
	Min.	Mean	Max.	Min.	Mean	Max.
Don't wait before transmit	140	155	170	310	326	342
Wait before transmit	220 + Tother	252 + Tother	285 + Tother	390 + Tother	423.5 + Tother	457 + Tother

II.6 - Communication Registers Definition

II.6.1 - Communication Registers

Several registers are used by communication programs. they are listed below :

Table 7 : Communication Register Definition

Register Name	Register Description
outstart	Output buffer start address
in_start	Input buffer start address
out_par	Parity byte to be send
rand_r	Random register
b_count	Byte counter
reg_delay	Delay before allowing carrier detect
portacopy	Port A data copy
portccopy	Port C data copy
start_sav	Begining of interrupt stack (down stack)
na_ctrl	communication control register (see below)
r_pa_reg	Register for received parity calculation
t_pa_reg	Register for transmit parity calculation
retry	Register for transmission retry(before aborting)
r_mess_r	Received message register
t_mess_r	Transmit message register
reg_trans	buffer for a single value
r_ad_r	Instantaneous address (read on switch)
t_dat_r	Transmitted data register
r_dat_r	Received data register
t_r_adr	Transmitted remote address
t_adr	Transmitted address

These 21 registers have a name, but more than 17 bytes are needed for communication. In fact, communication program needs 43 registers (that include software stack and buffer for communication).

II.6.2 - Network /Application Register

Application program is not sequential because of the use of several interrupts. These interrupts are used for reception and communication timing. The main program is able to know what are the communication status by looking at the Network/Application register.

The network/application register is used in two ways :

- To interface the application program with the communication program. The application program is able to know if order has been received or if an acknowledge has been received,
- To enable protocol control inside network program. That means byte error, unexpected frame, time-out, bad frame parity, transmit-enable.

Table 8 : NA_CTRL Definition

D0	D1	D2	D3	D4	D5	D6	D7
Byte error	Unexpected frame	Bad frame parity	Time-out	Transmit enable	Too many errors	frame acknowledge	order received

II.6.3 - Parity Register

Parity registers are used to compare received and transmitted parity frame value (last byte of frame). The parity registers have the same look, that is given below :

Table 9 : Parity Registers Definition

D0	D1	D2	D3	D4	D5	D6	D7
0	Pdata	Porder	Pda	Psa	Pctrl	Pha2	Pha1

- Pdatas : Parity of data byte
- Porder : Parity of order byte
- Pda : Parity of destination address byte
- Psa : Parity of Source address byte
- Pctrl : Parity of control byte
- Pha1 : Parity of first home address byte
- Pha2 : Parity of second home address byte

II.7 - Mains Flow

The ST6 must be able to receive a frame whenever it comes, but it is really important to leave CPU time for Application program. That's why the main is a loop, where the application program is running.
When a frame arrives, an IT occurs, the MCU receives the frame, modifies the communication status in Na_ctrl register and then go back to the main loop. If it is an order, the MCU send an acknowledge and does the order.

II.7.1 - Main Flow (see Figure 11)

When the board send a frame, it is waiting for an acknowledge, and tries 3 times if nothing is coming for a delay. If an acknowledge arrives, it is checked (parity check) and if a problem occurs, the MCU tries again (3 times maximum).The slave system main flow is the same as the master system, but of course, the application program changes.

Figure 11 : Main Program Organisation

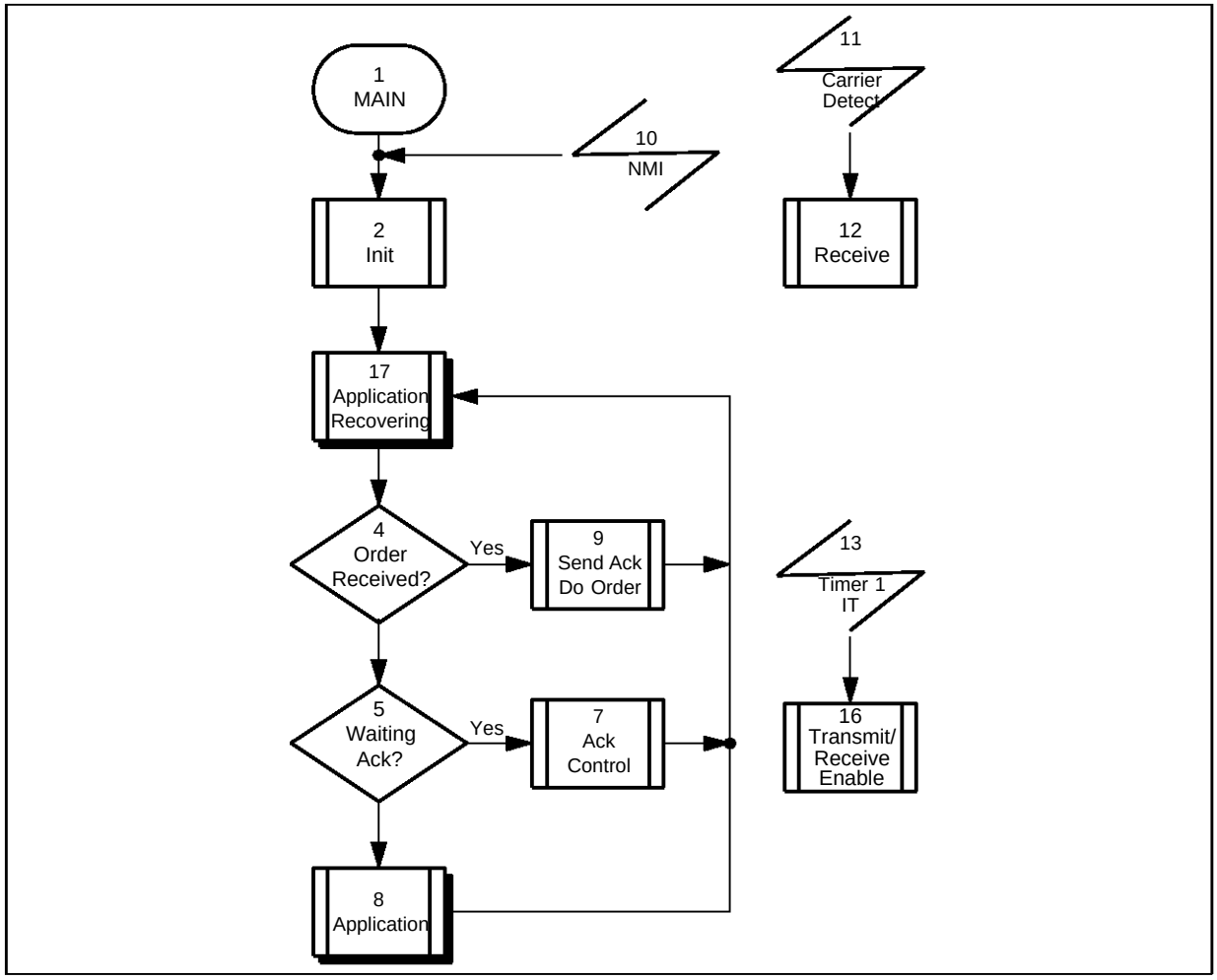
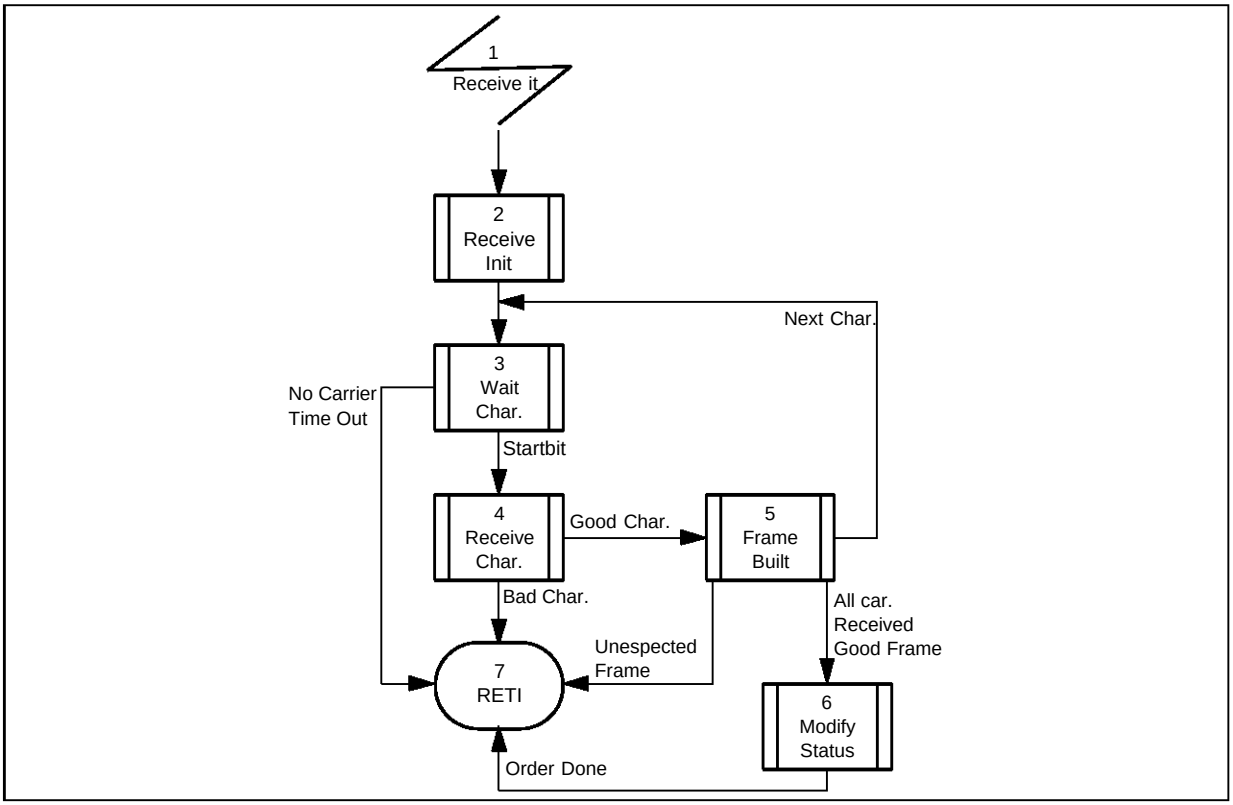


Figure 12 : Receive Routine



II.7.2 - Reception Flow

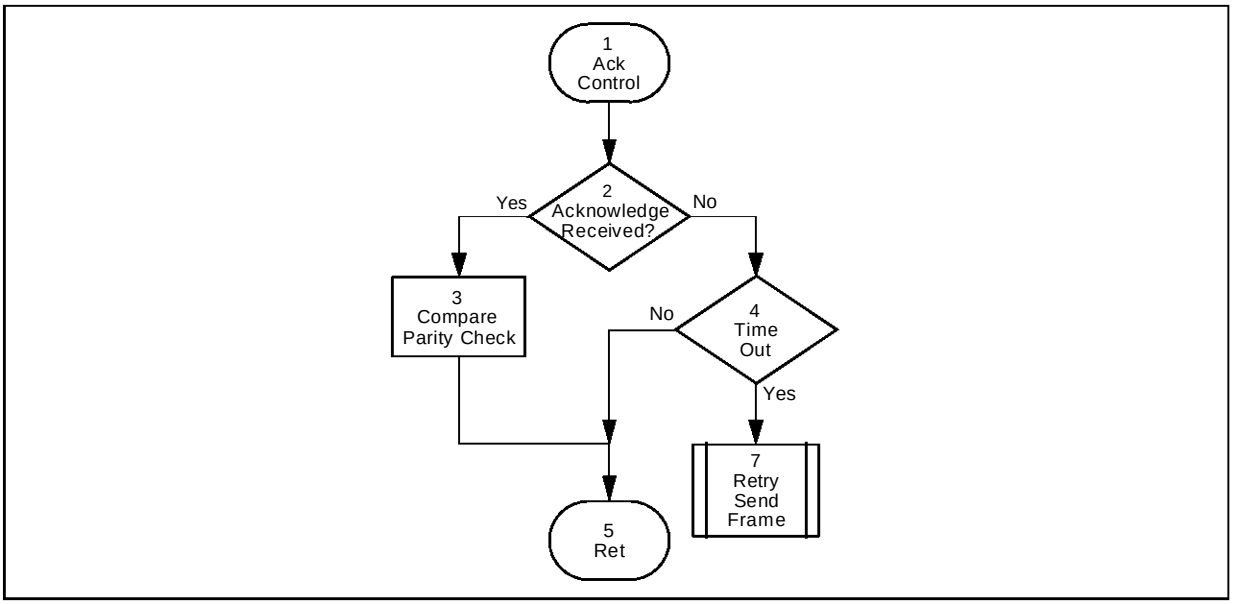
The synchronisation is done on each startbit : it's a byte synchronisation.

II.7.3 - Acknowledge Control Flow

The transmit part will not be detailed, it is just a pins and a timer control. The parity byte is calculated as the frame is sent, and it is sent after.

In order to leave MCU time for application, receive IT is disabled after each frame reception and for a delay of 30ms. This time is controlled by the Timer 1 and the reg_delay register.

Figure 13 : Ack Control Routine



III - APPLICATION : DIMMER CONTROL

III.1 - General Description (see Figure 14)

This section gives an example of an application that allows the dimming control of a remote light (or something else) by using the transmission with P.L.M. This application needs two boards :

- A master system, with a push-button (Dimming/off), 2 potentiometers and a 7 segments display,
- A slave system, with a 7 segments display and a plug for dimming control.

When the user presses the key (command), or turns a potentiometer, the master system sends a message, lights the decimal point on and waits an acknowledge. The remote board receives the message, sends an acknowledge and does the order. If no problem occurs, the master system receives the good acknowledge and switches the decimal point off. Otherwise, after trying sending the message 3 times, the decimal point remains lighting : a communication error occurs.

With the push-button, you can turn the light off or on. With the dimmer potentiometer, you can control the light intensity. With the display potentiometer, you change the number on the display. This feature has been added to prove that the ST6265 is able to receive and send order while it is dimming.

If you plug the master system in first, it will try to connect with the slave system three times, then it will light the error led (digital point of the 7 segments display). So you will have to plug the slave system and send an order (push the button and turn the display potentiometer) in order to return in normal mode.

The dimming control is done by the use of a triac (BTA08-600TW) so the load connected to the plug must not exceed 1000W.

III.2 - Schematics (see Figures 15 and 16)

This section gives the schematics of the two boards. pB7, pC0 and pA2 have different meaning on the master and on the slave board, that 's why we give two schematics. But it is possible to make a single board with dual implantation.

Warning : on this board the VDD is connected (via 0Ω resistor) to the neutral mains.

That means all the board is on the Mains supply voltage!

Figure 14 : Dimmer Control Application Description

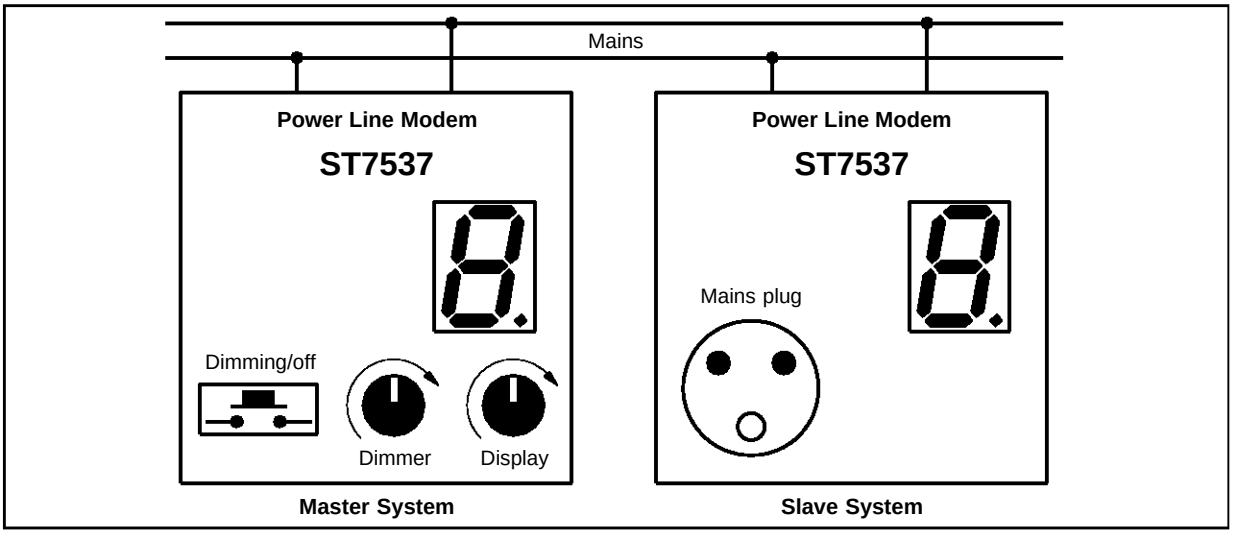


Figure 15 : Slave Board Schematic

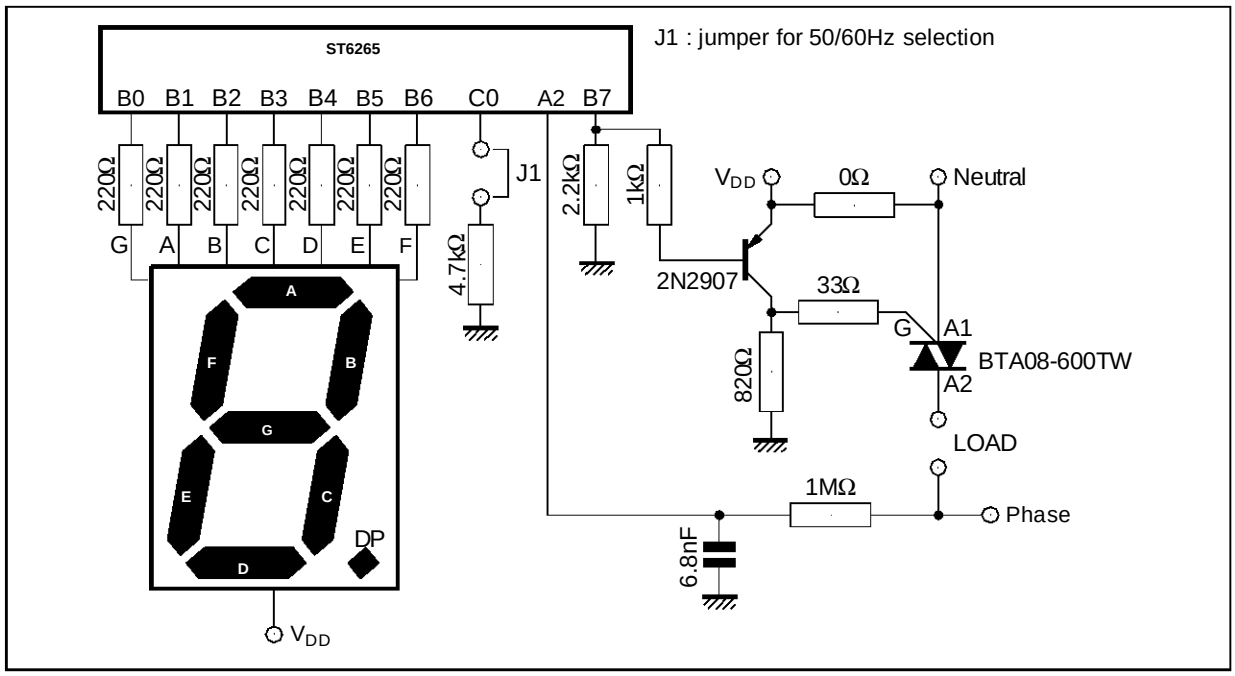
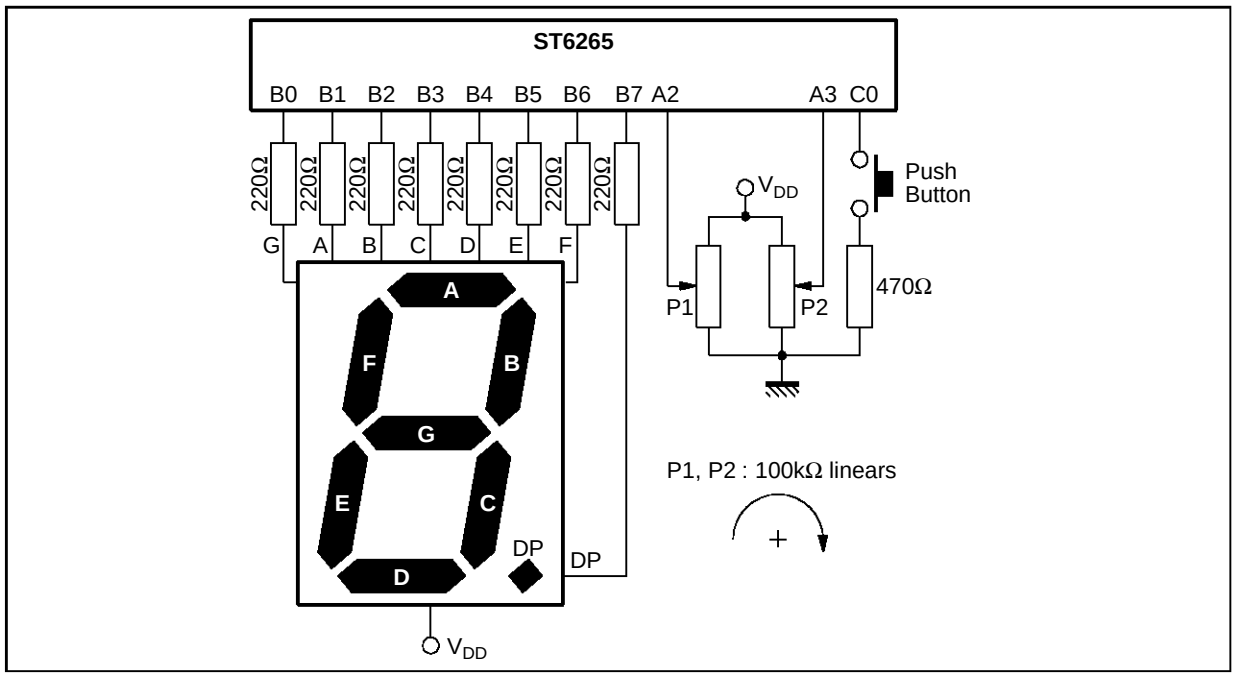


Figure 16 : Master Board Schematic



III.3 - Pins Description & Software Initialisation

III.3.1 - Slave Board

On this board, there is a 7 segments display application and a dimmer application.

The display application is very simple. The pins B0 to B6 have to be programmed in output with pull-up (push-pull). They are connected to the display by 220Ω resistors to limit the LED current. The display is a common anode so the pins have to be reseted to make the display lighting.

The dimmer application is most sophisticated, because it requires a zero crossing detection, and the possibility to wait a delay before firing the triac. See dimmer control part for more information.

The pin A2 is connected to the phase (via 1MΩ), in order to provide the zero crossing. This needs Neutral connected to V_{DD} to have a reference level. A2 has to be programmed in input.

The pin B7 has to be programmed in push-pull output to provide the pulse. B7 is the only pin that is connected to Auto Reload timer, so the fire signal needs to be amplified before driving the triac (the amplification stage is realised with a PNP transistor). At reset state B7 is configured in input with pull-up, so B7 is held at V_{SS} by a 2.2kΩ resistor.

This application is working with both 50Hz and 60Hz mains. The jumper connected to C0 is used to select between the two type of mains. Co has to be programmed in input with pull-up.

The pin configuration is the following :

Table 10 : Slave System Pins Configuration

Register	Bits Values	Byte Value (e.g.)	Description
DDRA	D2 res	00 h	A2 in input
ORA	D2 res	00 h	A2 in input
DRA	D2 set	04h	
DDRB	D0..D7 set	FF h	B0..B7 in output
ORB	D0..D7 set	FF h	B0..B7 in push-pull output set to 1
DRB	D0..D7 set	FF h	
DDRC	D0 res	00 h	C0 in input
ORC	D0 res	00 h	C0 with pull-up
DRC	D0 res	00 h	

III.3.2 - Master Board

On the master system, there are a display application and a control application :

- The display application the same as in slave system, but here, the decimal point is connected to B7. This decimal point is used as a warning light for communication acknowledgement. B0 to B7 have to be programmed in output push-pull,
- C0 is connected to a push-button, and has to be programmed in input with pull-up,
- A2 and A3 are input for analogic values coming from the potentiometers, so they have to be programmed in input, and reading an analogic value is done by programming one of these two pins in analog input, but not both in the same time.

Table 11 : Master System Pins Configuration

Register	Bits Values	Byte Value (e.g.)	Description
DDRA	D2 res D3 res	00 h	A2 in input A3 in input
ORA	D2 res/set D3 set/res	04 h /08 h	(A2 in input analog A3 in input) / (A3 in input analog A2 in input)
DRA	D2 set D3 set	0C h	
DDRB	D0..D7 set	FF h	B0..B7 in output
ORB	D0..D7 set	FF h	B0..B7 in push-pull output set to 1
DRB	D0..D7 set	FF h	
DDRC	D0 res	00 h	C0 in input
ORC	D0 res	00 h	C0 with pull-up
DRC	D0 res	00 h	

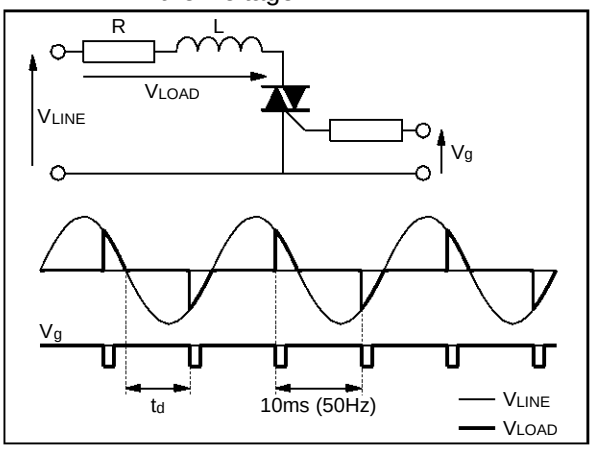
III.4 - Dimmer Control

III.4.1 - General Description

This section describes the main aspect of the power control system used on the slave system. For further details, refers to power control applications notes.

The output power is controlled by the phase delay of the triac drive. This delay is referred to the zero crossing of the line voltage. That needs an additional connection to main voltage, but in a remote application, this is not a problem.

Figure 17 : Power Control Based on the Monitoring of the Zero Crossing of the Voltage



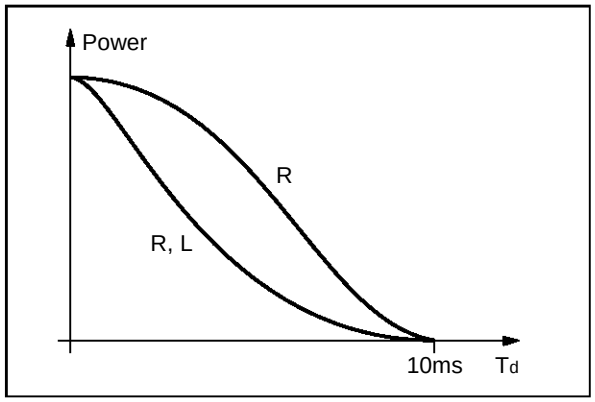
III.4.2 - Triac Specification

The triac used is a BTA08-600TW (SGS-Thomson) Logic Level triac. It has a maximum specified gate triggering current of 10mA (at 25°C). The current is provided by the amplifier stage (transistor), and the pulse width is programmed by software.

III.4.3 - Timing Specification

T_d determines the power used by the load. Between two pulses, there is a delay of 10ms, except during T_d variation.

Figure 18 : Load Power Consumption versus Time Delay



As shown in Figure 18, a small delay variation can affect the load power (that means light intensity in dimmer control), so this application is very sensitive to delay variations, and the timer has to be tuned very precisely. The timer is loaded with values stored in a table in ROM (one table for 50Hz, one for 60Hz).

III.5 - Application Software

III.5.1 - Display Application

The display routine is only checking the range of the transmitted data and setting the corresponding port B pins in order to display a digit. The display data is stored in the led_status register.

III.5.2 - Slave System

III.5.2.1 - AR Timer Configuration

The dimmer application is using the Auto Reload timer to provide the time delay (T_d). When a zero crossing voltage is detected on the mains voltage, the AR timer is launched with a delay (corresponding to T_d), And immediately after, the reload value is changed to a value corresponding to 10ms delay (50Hz). So, when an interrupt occurs (Receive mode for instance) the MCU is running in IT mode while the AR timer is providing the power control. The dimming intensity is controlled by the time delay, which has to be between Arr_min, and Arr_max.

The 10ms (or 8ms for 60Hz) delay is provided with the Arr_sd value. The Arcp value is loaded with Arrc + puls_siz ; this register defines the pulse width. All theses values are depending on the AR timer configuration :

- AR timer in autorelad mode with IT disabled : Armc = 20h
- AR prescaler set to 128 and clock division set to 3 : Arsc1 = 0E1h

Table 12 : AR Timer Configuration

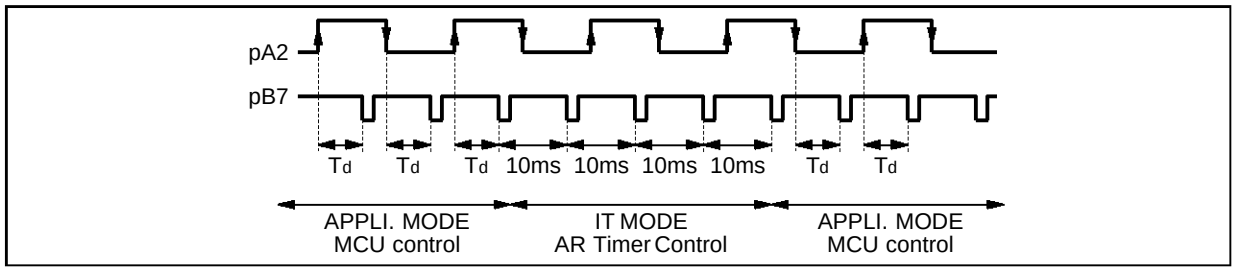
Register	50Hz Values	60Hz Values
Armc	20h	
Arsc1	0E1h	
Arrc during MCU control	Arr_min < Arrc < Arr_max	
Arrc during AR timer control	Arr_sd_10	Arr_sd_8
Arcp	Arrc + pulse_siz	

The values of Arr_min, Arr_max, Arr_sd are depending on the frequency of the MCU (Fint) and of the mains (50Hz or 60Hz). The delay waited between the zero crossing detection and the triac pulse is given by :

$T_d = T_{ar} \cdot (FFh - Arrc)$.

Where T_{ar} is the time for 1 count : T_{ar} = 6.98E-5s (at 5.5MHz).

Figure 19 : Timing Chart of Zero Cross Level Signal (pA2) and Triac Trigger Signal (pB7)



III.5.2.2 - Registers for Power Control

The application register is used to store the dimming status (on/off) and the zero crossing voltage.

Table 13 : Application Register Definition

D0	D1	D2	D3	D4	D5	D6	D7
lght_on	zero_dt	0	0	0	0	0	0

- lght_on : Set when dimming
- zero_dt : Set when pA0 is 0

Other bits are unused.

III.5.2.3 - Dimmer Flow

The dimmer procedure has to be launched very often because it provides dimmer synchronisation. If this procedure is not launched for more than 0.2ms, a test has to be performed in order to see if a zero crossing has appened during this delay. In this case, it is to late to synchronise on this zero crossing so the zero_dt bit of appli_reg has to be modified and the software will synchronise on the following zero crossing (Figure 20).

In the left flow : see Figure 20a
- Dimm_ctrl check if a new value has arrived for dimmer, calculate the new delay and eventually switch off or on the dimmer control,
- Aff_value checks if a new value has arrived for display, calculate the new digit and display it,
- get_level is a macro that get the level of the main voltage (1 for positive voltage and 0 for negative) and that modify the zero_dt bit of application register. This macro avoids delayed zero crossing detection.

In the right flow : see Figure 20b
- Zero crossing detection is done by comparing the instantaneous level (on pA2 pin) with the bit zero_dt in application register,
- AR timer control loads the arrc value corresponding to time delay, launch the timer, reload arrc value for a 10ms delay and arcp for the pulse width.

III.5.2.4 - Push-button

The detection of an action on the button is made simply by reading the value on pin C0. If somebody push on it the MCU sends a frame and waits for an acknowledge. This action needs more than 130ms so it makes a kind of debounce.

III.5.2.5 - Potentiometers

The main program reads the converted value on potentiometers, and compares it to the stored value (in dimm_val and in aff_val). If the difference between the read and stored value is higher than 8, the MCU send an order to remote system.

Figure 20 : Application Recover and Dimmer Procedure Flows

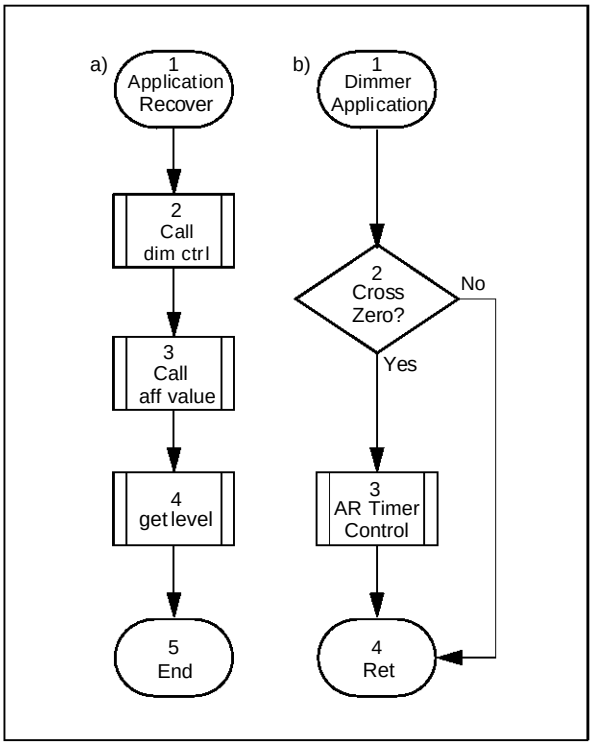
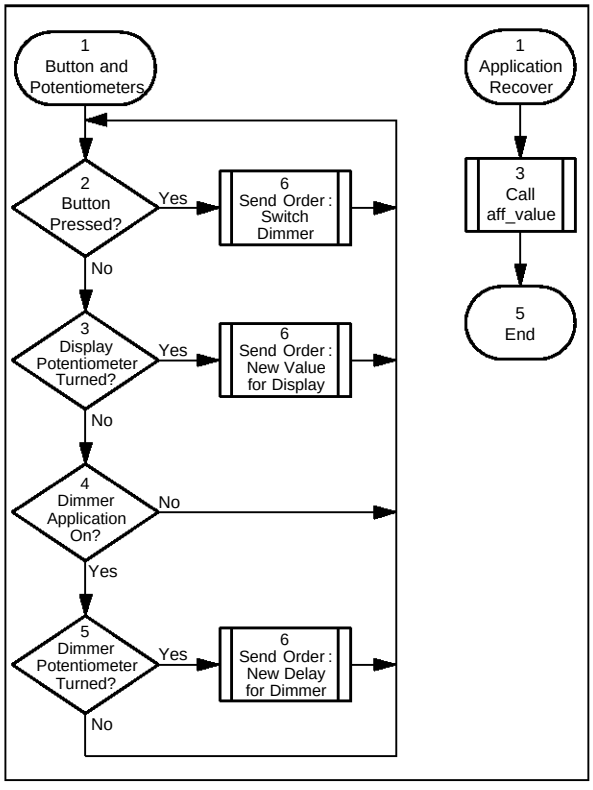


Figure 21 : Application Recover and Button and Potentiometer Control Procedure Flows



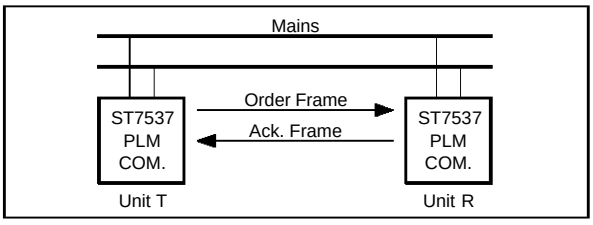
IV - NETWORK
IV.1 NETWORK SPECIFICATION

Power Line communication uses mains distribution cables that constitute a network adapted to the control of devices already connected on it. The ST7537CFN power line modem complies with the regulation described in the CENELEC EN50.065-1 document, (so frequency, bit coding and other useful features are included in the chip). Nevertheless, the designer of an application has to take in account the power line communication specification when writing his protocol. For instance, access protocols are required for coexistence on the medium. The following paragraphs give a concrete case of communication on power line network. In first, only 2 devices are connected to the network. Then, others devices are connected, and we will be confronted to a real network specification.

IV.1.1 - Communication between 2 Units

This is the simplest case of communication. Only two devices are able to send and receive packet on the network. One unit sends an order, the other one is replying with an acknowledge.

Figure 22



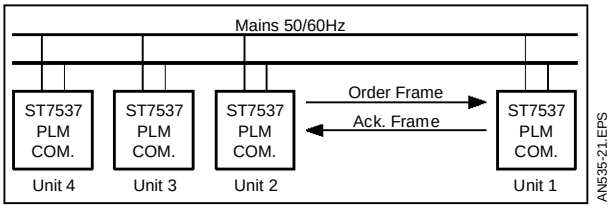
The unit T checks that the network is free for access thanks to the Carrier Detector, then it sends its frame and waits an acknowledge during a delay Twack. The unit R receive a frame and sends an acknowledge if R is the destination of the frame. R must send the ack. frame before the end of Twack.

If it is not the case, or if T has not received the acknowledge, T tries again to send the order. Of course the frame format must allows error detection by the receiver (order checksum) an by the transmitter (checksum in ack. frame). In most of the case, communication occurs without error, and the both units must keep silent for a delay Tsilent in order to leave time for application control. To allow the two units to transmit, the unit T has to wait more than R2, so R will be the first to take the channel if it needs to transmit.

If the two units send a frame exactly at the same time, they will not receive ack. frame, so they will retry after a delay (Twack). A random value is added to make one of the two unit faster than the other one. The total delay is Tretrans.

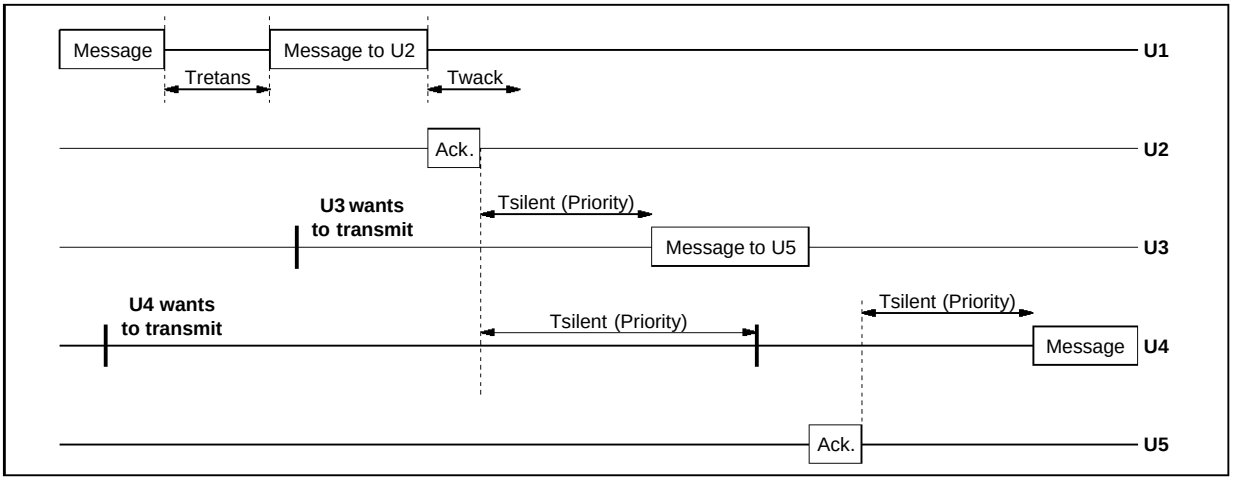
IV.1.2 - Communication with Several Units

Figure 23



With several units, the timing is the same, but even if a unit is not concerned with a communication, it has to get the frames in account for timing control. For instance, it has to reload its time to keep silent. And if several units have reload there time to keep silent at the end of a communication, the values reloaded must be different to avoid conflicts on the next communication. Again, a random value is added to make timing different.
Here is a data timing chart of the transmitted signal of the different units (see Figure 24).
Anyway, all units must send there message in less than two seconds.

Figure 24



IV.1.3 - Timing Control

All these timing are resumes in the following table.

Symbol	Description	Time	Comment
Tmax	Total duration of transmission	2s	
	Maximum duration of transmission after starting	1s	Feature included in ST7537CFN
TsilentR	Lenght to wait from the end of a remote transmission to initiate a transmission	85 .. 125ms	At least 7 values
Twack	Acknowledge sent after	0 .. 30ms	
Tretransmit	Retry transmitting	30 .. 72ms	
TsilentT	Duration between two transmissions of the same device	125ms	

In order to implement these timing, an easy way is to use a single timer and several registers corresponding to the different delays you want to count. The timer will decrease the registers at each overflow, and the counters are ”launched” by loading a value in the corresponding register. An example of implementation on ST6265 is given in part III.
This access protocol allows an additional network priority: if you allow unit 1 (U1) to transmit before unit 2 (U2), then U1 will always sends its messages before U2, and will have a highest network priority. By choosing the range of TsilentR of a unit, you will then choose its priority.

Values for TsilentR.

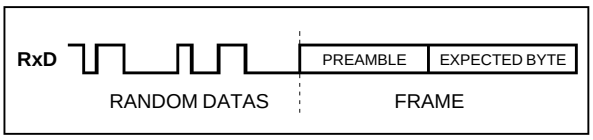
Range	Priority
85..94ms	Highest priority
95..104ms	Standard priority
105..115ms	Lowest priority

IV.2 - Example of Implementation of Soft Carrier Detector

We have seen that by programming the TxD to ”0” in receive mode we increase the sensitivity of the ST7537 because there is no more clamping by CD.

You will be able to have good communication with a receive signal of around 50dBµV which means a dynamic of around 70dB.
Because we want to get the benefit of the very good sensitivity of the ST7537, we will program Txd to ”0” in receive mode and create by soft a frame detector . We will use the CD signal as mentioned by CENELEC only when we want to transmit a frame . Different software frame detector can be implemented depending of the resources of your microcontroller. You can program your microcontroller to go in receive frame when it received the expected byte.

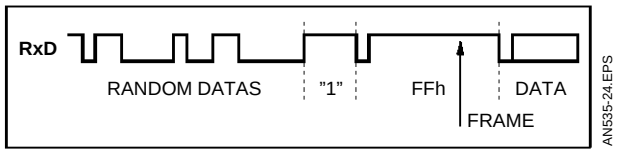
Figure 25



So the preamble is for demodulator training (when you start a communication the 3 first bits are lost by the receiver) and when you will match with expected byte the microcontroller will go in receive frame routine.

On the ST6 microcontroller we have implemented the following frame detector.

Figure 26



We put Txd =" 1" on the transmitter for around 4ms (for demodulator training) and

after we send in asynchronous mode FFh following by the complete frame. On the receiver , we check that we have RxD equal to "1" for at least 7ms (we are looking for FFh) , then we go in receive and we will have frame synchronisation on the first start bit of the data.

We did a trial in our lab with this system during 2 hours without having the ST6 going in frame receive routine on bad datas dued to noise signal.

IV.3 - Implementation of C.S.M.A on ST6265

The C.S.M.A. (Carrier Sense Multiple Access) needs a Timer for its implementation. But the ST6265 timer 1 is already used for bit time and software carrier detect. Furthermore, sometimes the timer has several functions at the same time, so the timer programming becomes very complicated.

In order to simplify this programming, we have implemented a single time delay corresponding to bit time (f = 1200Hz). So for each mode there is a counter corresponding to a delay. The counters are incremented (or decremented) in the timer interrupt routine while they are cleared (or affected with values) in main program.

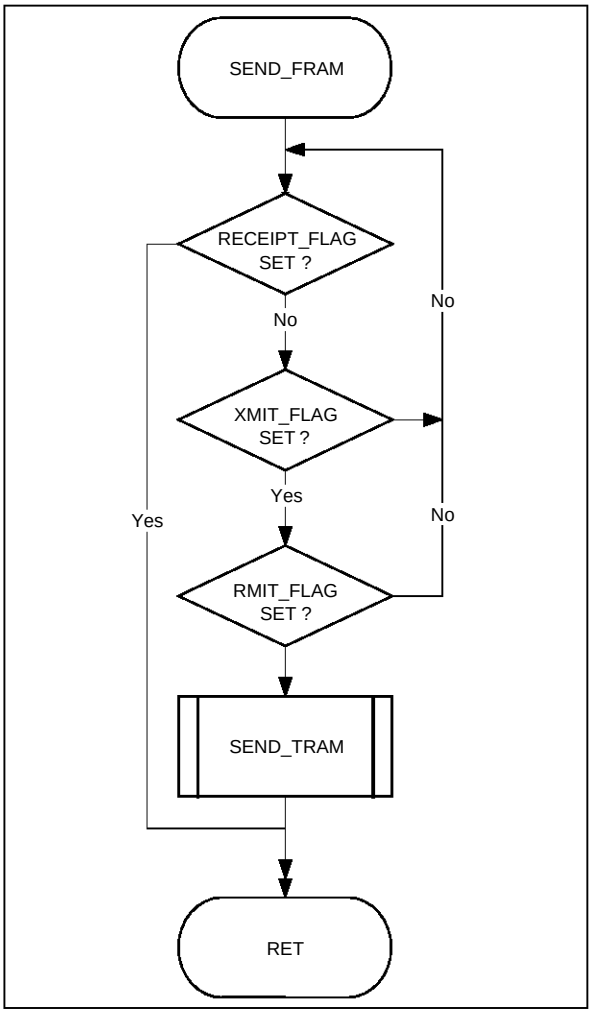
For C.S.MA. specifications, we use two counters :
- Xmit_count
  This is the delay before retransmitting
- Rmit_count
  This is the delay before transmitting after a reception

These counters are decremented in Timer 1 interrupt routine and flags are set when they become null. These flags are allowing the sending of a frame. Acknowledge frames are not concerned with these timing.

With this way of programming , the places where counters are loaded are very significant :
- Xmit_count is loaded at the end of the message sending procedure in order to wait an acknowledge (30ms) and at the end of the reception of a good acknowledge (time between two transmissions of the same device : 125ms).
- Rmit_count is loaded at the end of a reception with a random value (time between two transmission of different devices : 85 to 115ms).

The timer will allow the sending of a frame after C.S.M.A. delays.

Figure 27 : Sending a frame after C.S.M.A Delays



V - CONCLUSION

The ST7537 master and slave systems are demonstrative applications with low cost components. These boards are realised with discrete components but the size of these applications can be reduced by using a switching power supply and SMD (the ST7537 and the ST6265 are both available in small outline plastic). Several improvements should be done in order to make a more flexible product :

- Network management : All the addresses are fixed except the slave system address (switches). The software should be able to change the Home address, and objects addresses. This job needs management frame reception, emission and control,
- Byte transmission improvement : the software is reading the value on Rxd pin only one time (at the half of the bit), it would be better to do it several time to avoid spike perturbations,
- Frame transmission : the transmission is asynchronous with odd parity and there is a frame parity byte at the end of the frame, but if errors are detected, they are not corrected. A correcting code should be implemented in less than 200 bytes on ST6x,
- Power control : the 50Hz/60Hz detection should be done automatically instead of using a jumper. And the number of stages in dimming mode should be increased to obtain a pseudo-continuous variation.

All these modifications are realisable, because the program is less than 800 bytes long and the MCU is working at 5.5MHz, so there is place and CPU time left. The software is divided in module, so parts can be removed, and mains programs source (master.asm and slave.Asm) are less than 300 lines long.

VI - ANNEXE A : ST6x PROGRAMS

The program is divided in several modules, so it is easy to take parts of it and it is more readable. The master system and the slave system have common parts, but the mains programs are not the same.

Table 14 : Modules Used in Master and Slave Boards

Name	Function	System
6215_reg.asm	Common 62xx core registers	Master/Slave
ST6_7537.asm	Byte communication with ST7537	Master/Slave
Def_fram.asm	Frame control	Master/Slave
Address.asm	Address switches management	Master/Slave
Display.asm	7 segment display management	Master/Slave
Pot_et_b.asm	Potentiometers and push-button management	Master
Powerctl.asm	Power control	Slave
Master.asm	Main master program	Master
Dimmer.asm	Main slave program	Slave

The master and slave programs are compiled without linker, but when using the powerctl.asm module the option :
block 64% S64
is included in order to provide a good window banking for the tables in rom (AR values for dimmer application). That makes the object code bigger than it is but it is the price to pay.

VII - ANNEXE B : SCHEMATICS