7. Implementation of the CAN 2.0 protocol on HCS12 family

7.1. Purpose

Familiarity with serial communication protocol CAN 2.0 A / B, and understanding the configurations used in previous work, for sending a message on CAN bus using microcontroller MC9S12DJ256.

Skills required at the end of this lab::

- Determination of parameters that configure the duration of a bit on a CAN bus based on a transfer rate and the maximum length of wire communication;
- Registers configuration of the CAN module of HCS12 family of microcontrollers for access to a CAN bus with a transfer rate and a length of wire communication.

7.2. CAN communication protocol

Development of a CAN network began in 1983, when Bosch tried to replace communication systems "point-to-point" between control electronics device from vehicle, with a single serial bus to interconnect these controls elements. Although communication protocols were completed in 1986, the first circuits that integrate CAN controller are available from 1987. CAN version 2.0 used in HCS12 family of microcontrollers was published in 1991 (Bosch, 1991).

Some features of this protocol are support multicasting and broadcasting and the opportunity to change the number of devices connected to the network (nodes) dynamically without disturbing the operation of the other nodes. Throught CAN protocol, network nodes do not have associated specific addresses, identification of the receiver being achieved via identifiers of messages which also establish messages priority. In terms of physical access to the CAN network, are shown in Figure 34 elements used in previous works.

Figura 1 – Connecting devices to CAN network

The access to the physical level, represented by the two lines with resistance of 120 Ω, is made using CSMA / CA (Carrier Sense Multiple Access with Collision Avoidance) and NRZ encoding type (No Return to Zero). Thus, level communication environment is monitored continuously by each network node, any of them being able to start sending messages when there is no data flow. If multiple nodes try to send messages at the same time, only one, will continue the process and the remaining nodes appreciating collision will wait for completion of ongoing transmission. Two levels are defined for communication environment state, one dominant and one recessive. Access to the physical mechanism is of type AND-wired therefore recessive's level is equivalent to the logical value "1", and the dominant, is similar to logic level "0", so the physical environment is in recessive state only if all nodes transmit the value "1". Arbitration in case of collisions is done the non-destructive, at the bit level, a message is accepted by all nodes or none.

The structure of a message in extended format used in laboratory work includes:

- A bit of start with dominant level.
- 32 bits for arbitrary, of which 29 represent the message identifier in the extended format. This identifier is used for bus access arbitration, if two devices try to send messages simultaneously. Only the one with the smallest value of identifier will send the following fields of the message.

- 6 control bits of which 3 is the number of data bytes in the message. So the maximum length of the data field is 8 bytes.
- between 0 and 64 bits form a data field, the exact length is specified in the previous field.
- 16-bits represent field used for implementation of CRC mechanism.
- A 2-bits field used as confirmation request / response.
- 7 bits marks the end of a data message.
- 3 consecutive bits, recessive's, are used to separate messages.

Transfer rate in a CAN network is configurable by setting the quantum of time for a bit, and the number of quanta, as determined, each of the four segments that make up a bit: synchronization, propagation, phase 1 and phase 2. A detail of the relations between these segments, will be made upon presentation of the software configuration used.Synchronization segment is needed to introduce a window of time to attenuate the effect of the sliding for different synchronization signals used on CAN network nodes. Propagation segment reflects the signal delay due to its propagation through communication lines, so at longer distances between nodes, this segment of time will be longer. Reading the actual value of a bit is done between segments: phase 1 and phase 2.

7.3. S12MSCANV2/D Module

HCS12 family of microcontrollers are equipped with CAN module type S12MSCANV2 / D, this module documentation is available of the laboratory director (file S12MSCANV2-can.pdf). The structure of this module is described in Figure 35. The main element in this module is the control and state block , its operation is determined by the contents of the configuration registers. Through these configuration registers is selected source of synchronization signal used in determining the duration of time quanta, this being division used in determining the segments that make up a bit. Configuration registers which sets the duration of a bit and baud rate are CANBTR0 and CANBTR1. The parameters used in those two registers are:

- SJW: maximum number of time quanta that can altered a bit duration in resynchronisation step.
- BRP :division factor of the synchronisation signal to obtain quantum of time.

- SAMP: is an optional parameter, specifying line readings for the same bit at two time points in advance, the value finally considered being established on the principle of the majority.
- TSEG2: includes the number of quanta of time from propagation step and phase 1.
- TSEG1: Specifies the number of quanta of time after procurement moment of the bit value (phase 2).

The settings necessary to establish duration of a bit at a bit rate of 500 kbps (configuration used in laboratory work) is determined by taking the following steps:
1) duration of a bit is $t = 1s \cdot 500.000^{-1}$ bits/s $= 2 \cdot 10^{-6}$s;
2) $t = \frac{BRP}{f_{CANclk}}(\mathrm{SJW} + \mathrm{TSEG1} + \mathrm{TSEG2})$;
3) each bit is composed of four time segments, each segment being a multiple of quantum tq: synchronization, propagation, phase 1 and phase 2;
4) bit reading is between the last two segments (phase 1 and phase 2) so their duration must to respect simultaneously conditions: $8t_q \leq t \leq 25t_q$ şi $2t_q \leq$ phase $2 \leq 8t_q$ şi $4t_q \leq$ propagation + phase $1 \leq 16t_q$ şi SJW $\leq 4t_q$ şi SJW $\leq$ min[phase 1, phase 2];

Figura 2 – Block diagram of the module MSCAN (Motorola, 2004)

5) Propagation segment has the role of compensation for delays due to signal propagation on the CAN bus. The usual values are between 1 and 14 tq according to the length of the cables. Given the fact that at a rate of 500 kbps line length can reach several hundred

meters, and the working point is place on approximately 2 m2, so a restricted area, you can choose a small value for this time (2 TQ);
6) Typically, the ratio between the frequency reference signal and its prescaling factor is an even number and the values for the number of quanta in Phase 1 and Phase 2 are chosen so as to satisfy one of the relations phase 2 = phase 1 or phase 2 = phase 1+1;
7) If SJW = 1 t_q, propagation = 2 t_q, f_{CANclk} = 16MHz and if a prescaling factor is chosen BRP = 4, conform according to the above conditions, the minimum values are obtained : phase 1 = 2 t_q and phase 2 = 3 t_q.

7.4. Conducting work

1. Defining the requirements of the work

   Familiarize with code structure associated to CAN bus communication.

2. Open the CodeWarrior project related to one of the previous work, namely .mcp file. Track contents of mscan.h and m stan.c file ; data structures used, and settings made in *CANInit* function are taken into account:

```
// ##################################################################
void CANInit(unsigned char can_num){
unsigned char *can_pt;
can_pt = can_periph[can_num];
// set INITRQ, this will also set INITAK
can_pt[R_CANCTL0] |= M_CANCTL0_INITRQ;
// wait for init mode to occur
while (!(can_pt[R_CANCTL1] & M_CANCTL1_INITAK));
// Set CANE just in case this is the first time after reset
can_pt[R_CANCTL1]  = M_CANCTL1_CANE;
can_pt[R_CANBTR0]=0x43;
can_pt[R_CANBTR1]=0x23; // for a CAN Baud of 500kbps at 16Mhz crystal
// Disables all the Filters
   can_pt[R_CANIDMR_1B+0] = 0xFF;
   can_pt[R_CANIDMR_1B+1] = 0xFF;
   can_pt[R_CANIDMR_1B+2] = 0xFF;
```

```
    can_pt[R_CANIDMR_1B+3] = 0xFF;
    can_pt[R_CANIDMR_2B+0] = 0xFF;
    can_pt[R_CANIDMR_2B+1] = 0xFF;
    can_pt[R_CANIDMR_2B+2] = 0xFF;
    can_pt[R_CANIDMR_2B+3] = 0xFF;
  can_pt[R_CANIDAC]=0; // set filters to 2 32 bit acceptance filters
  can_pt[R_CANCTL0] &= ~M_CANCTL0_INITRQ; // clear INITRQ
  while (can_pt[R_CANCTL1] & M_CANCTL1_INITAK);
  // Set CANE just in case this is the first time after reset
  can_pt[R_CANCTL1]  |= M_CANCTL1_CANE;
  can_pt[R_CANRIER]  |= 1; // enable receive interrupt
  }
```

3. Will be analyze the CAN module configuration of *CANInit* function based on available resources: CAN documentation: S12MSCANV2-can.pdf, INT interrupt module specification, S12INTV1-int.pdf, related documentation of ports and HCS12 configuration: S12DP256PIMV2-portguide.pdf , 9S12DP256BDGV2-userguide.pdf and ZK-S12-b_schematic.pdf.
4. Configure the values for transfer rates:
   a) 1Mbps
   b) 54 kbps
   c) 100 kbps

7.5. Commented references

- Related Data Sheet to MSCAN module : S12MSCANV2-can.pdf (Motorola, 2004) presents configurations and CAN module functions integrated on S12 microcontrollers.
- Official Data Sheet of CAN 2.0 protocol : CAN Specification 2.0, Robert Bosch GmbH, September 1991

- Related Data Sheet to HCS12 family HCS12 : 9S12DP256BDGV2-userguide.pdf (Freescale, 2005) presents the different functions of microcontroller terminals.
- Related Data Sheet to microcontroller ports : S12DP256PIMV2-portguide.pdf (Freescale, Port Integration Module, 2002) presents a description of the various ports registers.
- Related Data Sheet to interrupt module INT of HCS12 microcontroller: HCS12: S12INTV1-int.pdf (Freescale, Interrupt Module, 2003) presents description of this module.
- Connection to the board level of ZK-S12-B is presented in zk-s12-b_schematic.pdf (Freescale, ZK-S12-B, Schematic and Bill of Material).

7.6. Worksheet

Name, Surname:……

Semigroup:……..

1. Explain the following configurations:

CANBTR0=0x43:……

.....……

CANBTR1=0x23:……

.....……

2. What reference signal is used in previous works to determine the quantum of time on CAN bus, CPU- clock or bus-clock ? Arguments response under the Code settings.

.....……

.....……

.....……

.....……

3. Configure the following registers for a 1 Mbps baudrate at an 8MHz clock reference:
CANBTR0=……

CANBTR1=……

4. Configure the following registers for a 1 Mbps baudrate at an 16MHz clock reference:
CANBTR0=……

CANBTR1=……

5. Describe the mechanism by which a received message on the CAN bus comes to be treated in the main loop of the function main () in the code used in previous works. Details of the events that occur, where they are treated and the route information in the data field of a CAN message.

.....……

.....……

.....……

.....……