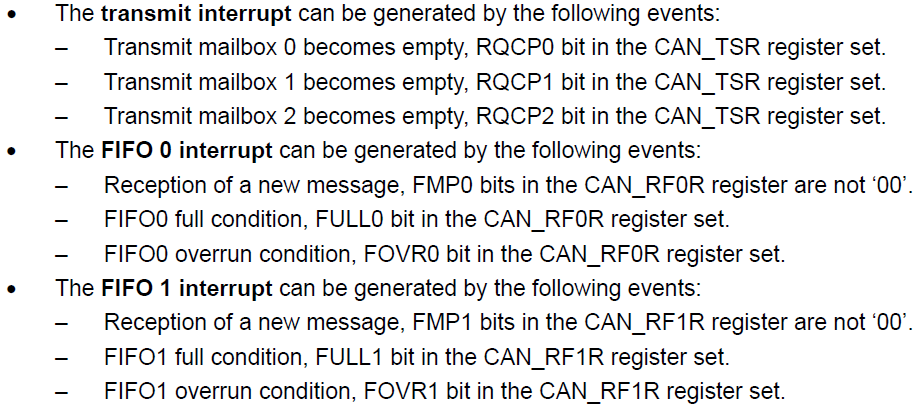
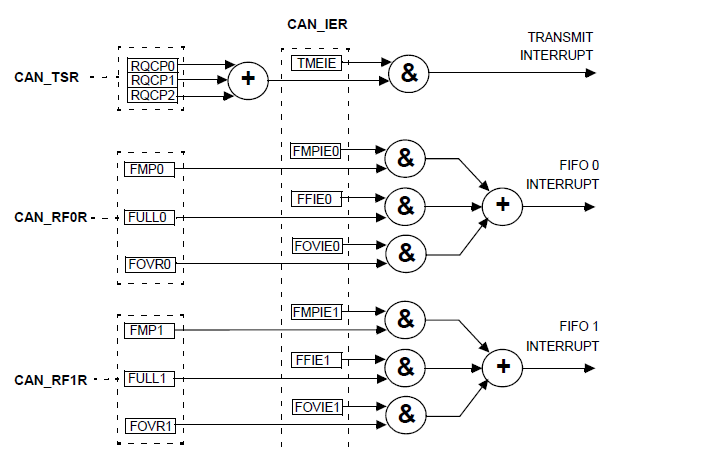
**CAN Interrupts**



**CAN Initialization mode**

**The software initialization can be done while the hardware is in Initialization mode.**

To enter this mode :

the software sets the INRQ bit in the CAN\_MCR register and waits until the hardware has confirmed the request by setting the INAK bit in the CAN\_MSR register.

To leave Initialization mode:

the software clears the INRQ bit. bxCAN has left Initialization mode once the INAK bit has been cleared by hardware.

While in Initialization Mode, all message transfers to and from the CAN bus are stopped and the status of the CAN bus output CANTX is recessive (high).

Entering Initialization Mode does not change any of the configuration registers.

To initialize the CAN Controller, software has to set up the Bit Timing (CAN\_BTR) and CAN

options (CAN\_MCR) registers.

To initialize the registers associated with the CAN filter banks (mode, scale, FIFO

assignment, activation and filter values), software has to set the FINIT bit (CAN\_FMR). Filter

initialization also can be done outside the initialization mode.

*Note: When FINIT=1, CAN reception is deactivated.*

*The filter values also can be modified by deactivating the associated filter activation bits (in*

*the CAN\_FA1R register).*

*If a filter bank is not used, it is recommended to leave it non active (leave the corresponding*

*FACT bit cleared).*

**void can\_enter\_init()**

**{**

**sbi(CAN1->MCR,0); //Enter into initialization mode by setting INRQ in MCR**

**cbi(CAN1->MCR,1); //Exit from Sleep mode by clearing SLEEP bit as its auto SET at Reset**

**sbi(CAN1->MCR,4); //No Retransmit, Msg will be Transmitted Once**

**while(!(cb(CAN1->MSR,0))); //normally it is 0 when it set-->initialization mode**

**while(cb(CAN1->MSR,1)); //in Sleep mode SLAK=1 when SLAK=0-->exit from sleep mode**

**}**

