Preparation:

Q.1:

```
extern ARM_DRIVER_CAN Driver_CAN1;

Driver_CAN1.Initialize(NULL,NULL);
Driver_CAN1.PowerControl(ARM_POWER_FULL);
Driver_CAN1.SetMode(ARM_CAN_MODE_INITIALIZATION);
Driver_CAN1.SetBitrate(

ARM_CAN_BITRATE_NOMINAL, 125000,
ARM_CAN_BIT_PROP_SEG(5U) |
ARM_CAN_BIT_PHASE_SEG1(1U) |
ARM_CAN_BIT_PHASE_SEG2(1U) |
ARM_CAN_BIT_SJW(1U));

Driver_CAN1.ObjectConfigure(0,ARM_CAN_OBJ_RX);
Driver_CAN1.SetMode(ARM_CAN_MODE_NORMAL);
```

Q.2:

```
extern ARM_DRIVER_CAN Driver_CAN2;

Driver_CAN2.Initialize(NULL,NULL);
Driver_CAN2.PowerControl(ARM_POWER_FULL);
Driver_CAN2.SetMode(ARM_CAN_MODE_INITIALIZATION);
Driver_CAN2.SetBitrate(

ARM_CAN_BITRATE_NOMINAL, 125000,
ARM_CAN_BIT_PROP_SEG(5U) |
ARM_CAN_BIT_PHASE_SEG1(1U) |
ARM_CAN_BIT_PHASE_SEG2(1U) |
ARM_CAN_BIT_SJW(1U));

Driver_CAN2.ObjectConfigure(1,ARM_CAN_OBJ_TX);
Driver_CAN2.SetMode(ARM_CAN_MODE_NORMAL);
```

Q.3: Dans cet cas on peut utilizer une table avec des filtres pour des identifiants specifiques.

```
Driver_CAN1.ObjectSetFilter( 0, ARM_CAN_FILTER_ID_EXACT_ADD ,
ARM_CAN_STANDARD_ID(0x161), 0) ;

Driver_CAN1.ObjectSetFilter( 0, ARM_CAN_FILTER_ID_EXACT_ADD ,
ARM_CAN_STANDARD_ID(0x0b6), 0) ;
```

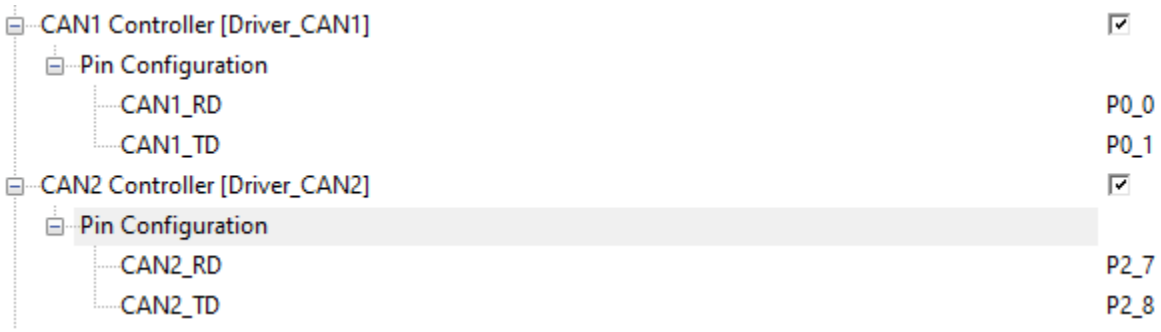
Q.4:

```
ARM_CAN_MSG_INFO tx_msg_info;

tx_msg_info.id = ARM_CAN_STANDARD_ID (0x0b6);
tx_msg_info.rtr = 0; // 0 = trame DATA
data_buf [0] = 0xFA; // data à envoyer à placer dans un tableau de char

Driver_CAN1.MessageSend(1, &tx_msg_info, data_buf, 1);
```

https://drive.google.com/drive/folders/1iqufj9sVkTgJt7lpXY9O_f5XWARdJt3G

https://drive.google.com/drive/folders/1iqufj9sVkTgJt7lpXY9O_f5XWARdJt3G

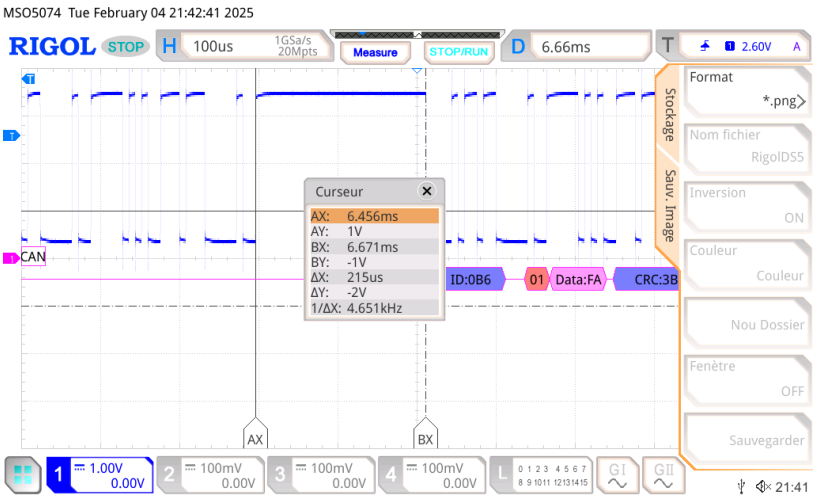
Q.5:



Q.6:

```
void CANthreadT(void const *argument)
{
  ARM_CAN_MSG_INFO                tx_msg_info;
  uint8_t data_buf[8];

  tx_msg_info.id = ARM_CAN_STANDARD_ID (0x0b6);
  tx_msg_info.rtr = 0; // 0 = trame DATA
  data_buf [0] = 0xFA; // data à envoyer à placer dans un tableau de char


  while (1) {

    // Code pour envoyer trame Id 0x0f6
    Driver_CAN2.MessageSend(1, &tx_msg_info, data_buf, 1);
    //.............

    osSignalWait(0x01, osWaitForever);     // sommeil en attente fin emission
    osDelay(100);
  }
}
```

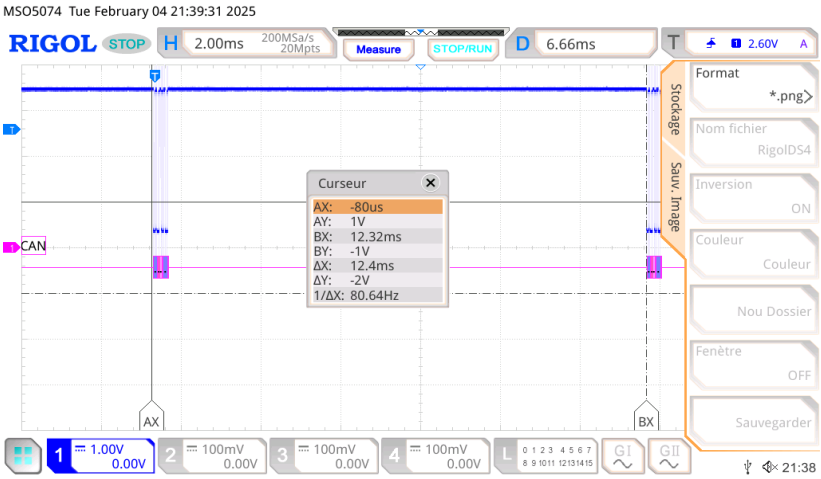
Q.7:



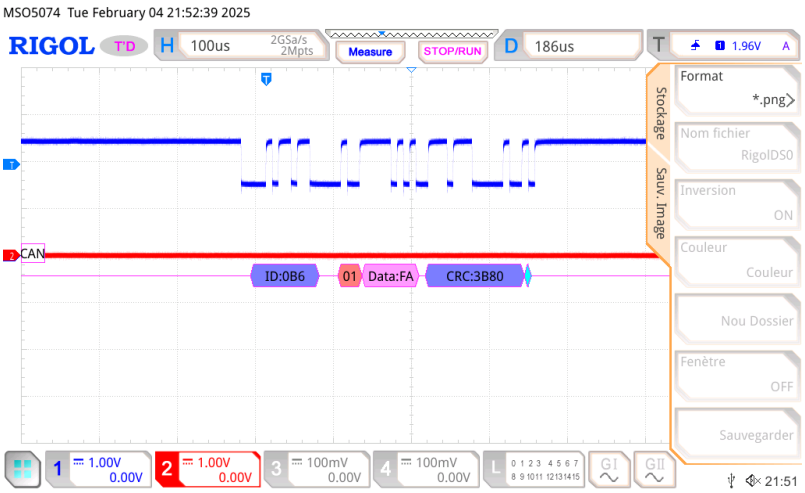
100ms = 10 Hz,

Ici, la trame est envoyé en permanence car il n'y à pas un 'ACK' pour indiquer la reception.

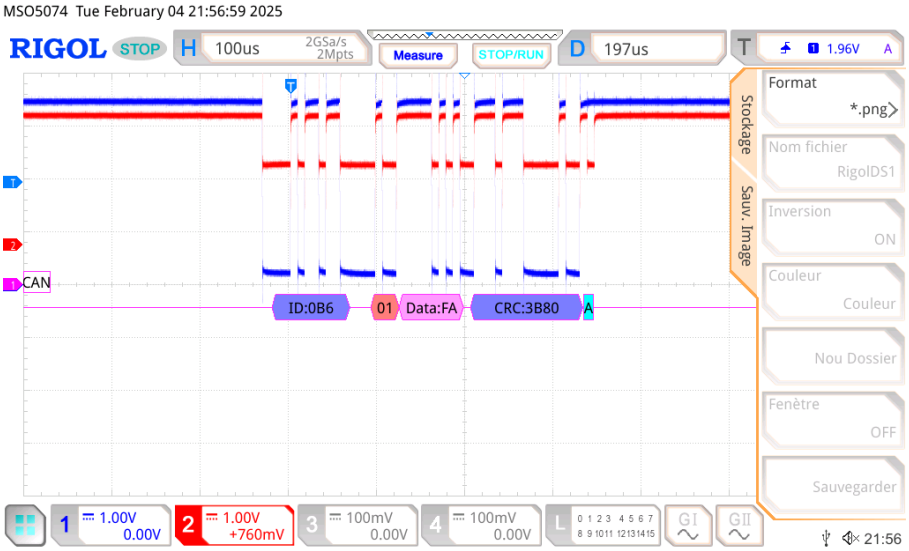
Q.8:



Mainenant, la trame est envoyé periodiquement.



Q.9:

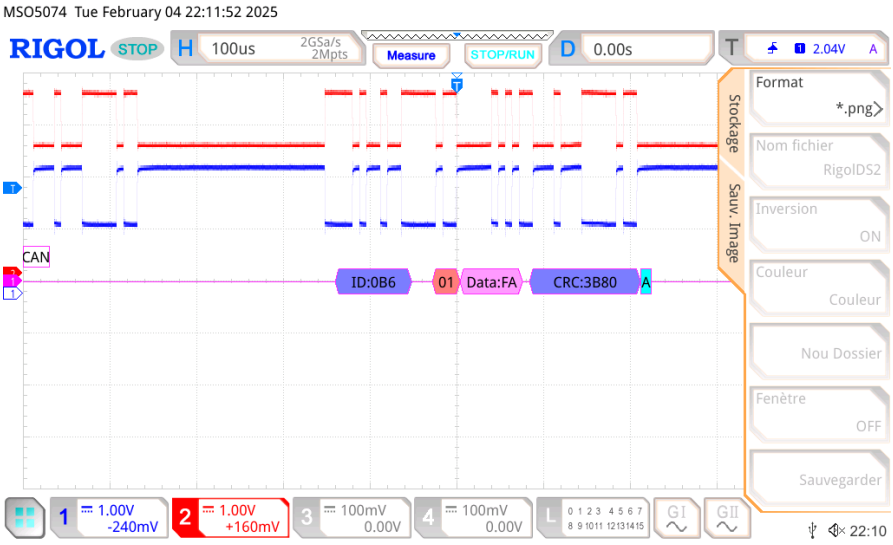


Ici CH1 = P2.8 et CH2 = Tx2

On peut voir le 'ACK' depuis le CAN1 qui vient d'ecraser la valeur récessife de CAN2.

(dernier bit a 1 sur CH1, mais 0 sur CH2)

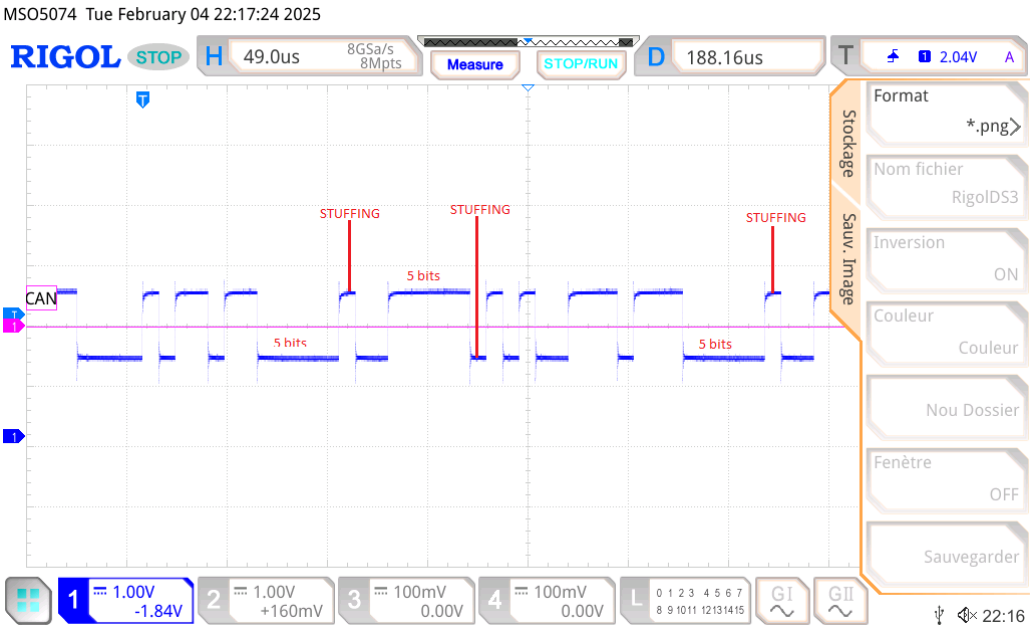
Q.10:



CH1 = CAN_L

CH2 = CAN_H

Q.11:



Q.12:

Q.1:

```
// tache envoi toutes les secondes
void CANthreadT(void const *argument)
{
	ARM_CAN_MSG_INFO		tx_msg_info;
	uint8_t data_buf[8];

	tx_msg_info.id = ARM_CAN_STANDARD_ID (0x0b6);
	tx_msg_info.rtr = 0; // 0 = trame DATA
	data_buf [0] = 0xFA; // data à envoyer à placer dans un tableau de char


	while (1) {

		// Code pour envoyer trame Id 0x0f6
		Driver_CAN2.MessageSend(1, &tx_msg_info, data_buf, 1);
		//.............

		// osSignalWait(0x01, osWaitForever);		// sommeil en attente fin emission
		osDelay(100);
	}
}
```

Q.1:

Q.1:

Q.1: