life.augmented

ISEN | école d'ingénieurs
ALL IS DIGITAL!

STM32 NUCLEO

Périphérique I2C

ISEN | école d'ingénieurs
ALL IS DIGITAL!

STM32 Nucleo Board

Documentations techniques utiles

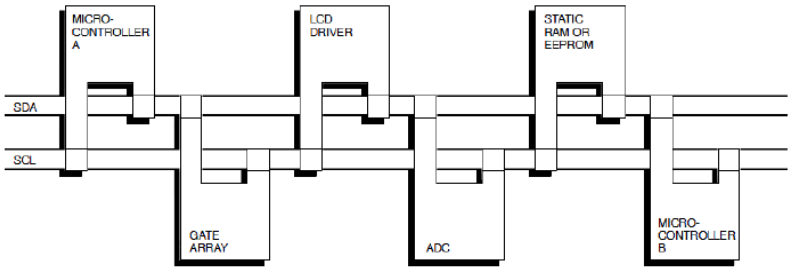
PM0056 Programming manual
STM32F10xxx/20xxx/21xxx/L1xxxxCortex-M3 programming manual

RM0038 Reference manual
STM32L100xx, STM32L151xx, STM32L152xx and STM32L162xx advanced ARM-based 32-bit MCUs

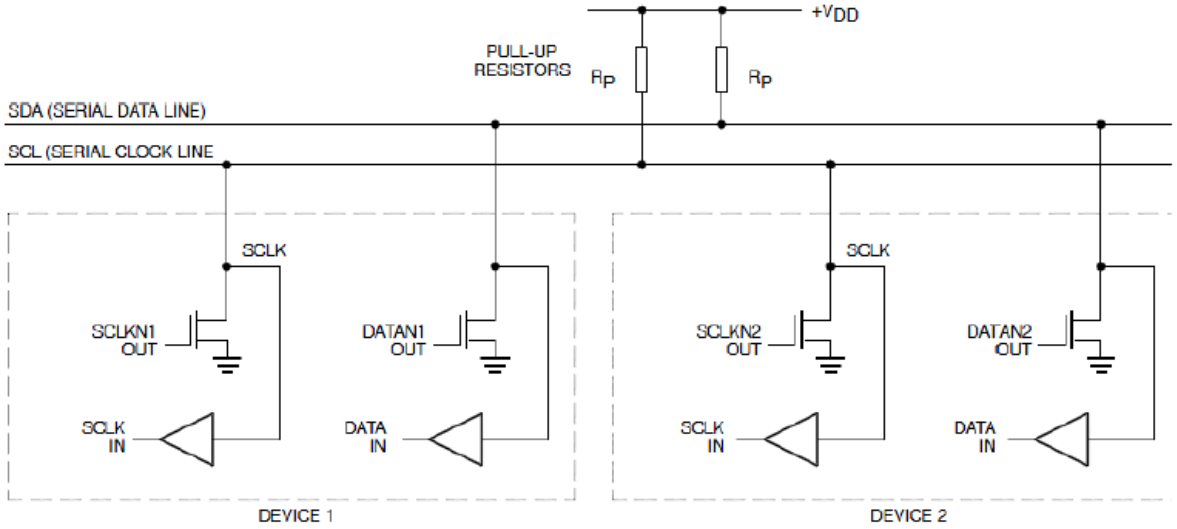
I2C : *IIC : Inter-IC-Communication*

- C'est un protocole d'échange courte distance, synchrone, bidirectionnel et half-duplex
- Inventé par Philips en 1982 !
  - Spécification 1,0 en 1992 – Vitesse maxi 100 KBits/s
  - Repris par NXP. La Dernière évolution de la spécification v 6.0 date de 2014.
- Les échanges se font entre un (ou plusieurs) Maître/Master et un (ou plusieurs) Esclave/Slave
  - Toujours à l'initiative d'un maître, pour un esclave.
  - Les données échangés sont sur 8 bits
- Il existe 5 vitesses de transmission (Fréquence d'horloge)
  - Standard Mode : Jusqu'à 100 Kbits/s
  - Fast Mode : Jusqu'à 400 Kbits/s
  - Fast Plus Mode : Jusqu'à 1 Mbits/s
  - High-Speed Mode : Jusqu'à 3,4 Mbits/s
  - Ultra-Fast Mode : Jusqu'à 5 Mbits/s

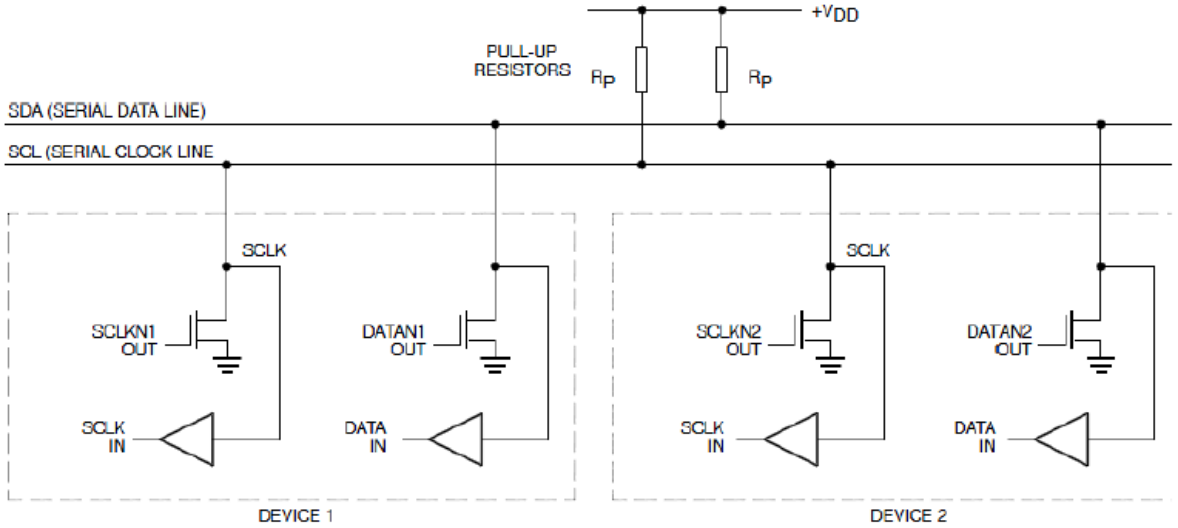
- Physiquement, le bus est constitué de deux fils (plus une référence de tension) :
  - **SCL** : serial clock qui est l'horloge de cadencement des communications
  - **SDA** : serial data permet les échanges bidirectionnels entre le maitre et un esclave
- Le bus est piloté par un maître (MASTER) qui génère l'horloge SCL de communication (il est possible de travailler en mode multi- maître).
- Tous les autres circuits sont esclaves (SLAVES), ils reçoivent tous la même horloge issue du maître.
- Chaque esclave possède une adresse unique sur le bus
- Maîtres et esclaves sont connectés sur le même bus au travers des mêmes fils SCL et SDA



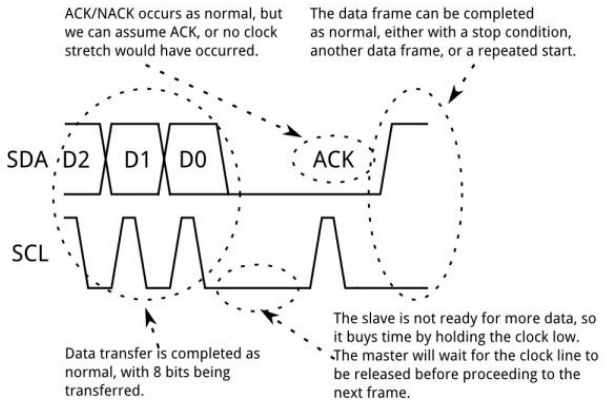
- Comment connecter des sorties entres elles ?
  - Les interfaces de sorties SCL et SDA sont constituées de transistor MOS en Open-Drain pour éliminer les court-circuit électrique !
  - Il faut des PULL-UP externes sur ces 2 fils
  - La valeur des PULLUP dépend de la vitesse que l'on veut atteindre (temps de montée)
  - Notion d 'état Dominant (le 0) et de Récessif (le 1).



- Comme entrée et sortie se font sur un même fil, un circuit peut vérifier l'état récessif de la ligne. Il place un 1 logique (transistor bloqué) et vérifie que la ligne est bien à 1, dans le cas contraire, c'est qu'un autre circuit est en train de placer un 0. L'émetteur peut donc vérifier l'émission effective de chaque bit.

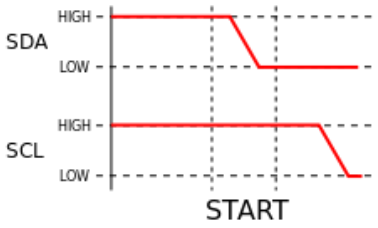


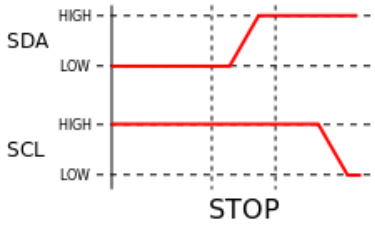
- Notion de pause (stretching en anglais)
  - Permet à un esclave de signifier au maître qu'il n'est pas prêt pour recevoir le byte suivant.
  - Il force alors un 0 sur la ligne SCL
  - Le maître qui voit ce 0 sur la ligne attend que le slave relâche l'action sur le SCL pour continuer sa transmission !

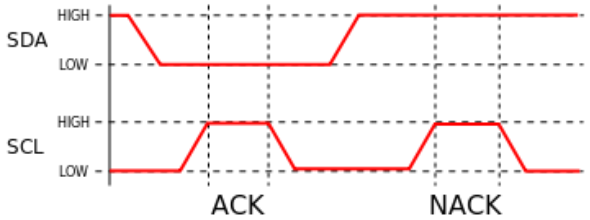


- Le Codage des bits se fait en NRZ (Non Return to Zero)
  - Pour faire simple ici :
    - Le Bit ‘1’ est codé par une tension proche de +VCC
    - Le Bit ‘0’ est codé par une tension proche de 0

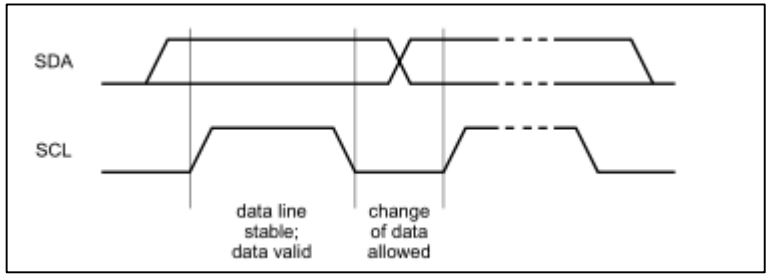
- Le START / Le STOP / Le ACK / Le NACK
  - Le START permet de commencer une transaction
  - Le STOP permet de la terminer
  - Le ACK permet d'acquitter la donnée
  - Le NACK permet de refuser la donnée ou signifier la fin d'une lecture







- Le bit de DONNEE
  - La donnée sur SDA doit être stable sur le niveau HAUT de SCL



- L’adressage
  - Chaque esclave est identifié par son adresse
    - Soit sur 7 bits (Cas le plus répandu).
    - Soit du 10 bits
  - L’adresse permet à l’esclave de savoir si on lui parle
  - Certaines adresses sont réservées : 2 Exemples parmi N
    - « 0000000 » : permet d’adresser tous les esclaves (broadcast)
    - « 11110xy- » : réservée pour l’adressage 10 bits
      - X et y représentent les 2 premiers bits de l’adresse 10 bits
      - L’octet suivant permettra d’envoyer les 8 bits restant

MAITRE
ESCLAVE

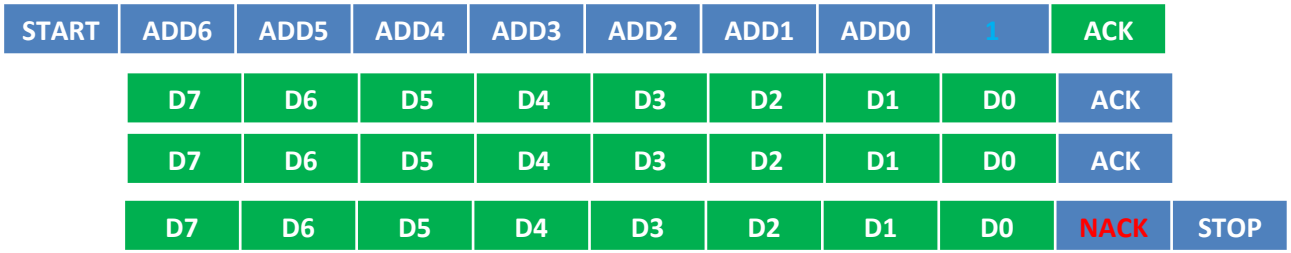
- Le protocole I2C : (Exemple avec adresse 7 bits)
  - Une transaction commence toujours par un START et finit par un STOP
  - Suivie de l'adresse du slave avec qui l'on veut « échanger »
  - La première donnée permet d'envoyer l'adresse de l'esclave sur 7 bits plus un bit R/W - Read ou Write (Read = 1 et Write = 0) : on a bien 8 bits !!!

START	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0	R/W	ACK	XXXXXXXXXXXXXX	STOP

- L'esclave qui se reconnait (ADD6/ADD0 c'est son adresse) fait un ACKnoledge
- Un échange Lecture/Ecriture (en fonction du bit **R/W**) va alors avoir lieu entre le maitre et l'esclave sélectionné
- Notion d'adresse I2C 8 bits
  - Soit un circuit I2C dont l'adresse sur 7 bits est 1010000.
  - On dit en général que l'adresse I2C (sur 8 bits donc) sera paire 10100000 (0xA0) en écriture et impaire 10100001 (0xA1) en lecture.

MAITRE
ESCLAVE

- Exemple Lecture de 3 bytes
  - Notez bien le NACK que doit renvoyer le maitre sur le dernier byte !! Pour signifier a l'esclave qu'il ne veut plus de données.



- Exemple d'écriture de 2 bytes
  - L'esclave ne fait que renvoyer des ACK (éventuellement fait du stretching sur le ligne)

Registres I2C STM32

- Deux types de registres
  - Registres de contrôle permettent de paramétrer/controler la macrocell I2C
    - Choisir le mode Maitre ou esclave, Envoyer un START, Envoyer un STOP, Envoyer un ACK, Envoyer un octet, Lire un octet, Paramétrer la vitesse, etc …

START	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0	1	ACK
CTRL	CTRL								ACK

  - Registres de statuts permettent de connaitre l'état de la macrocell I2C afin de cadencer les échanges en respectant les timings I2C et en gérant les erreurs de communications éventuellement
    - Le START est-il bien parti sur la ligne, Le Byte est t-il bien parti sur la ligne, Interruption, Erreur de communication, etc …

Registres de contrôle I2C STM32

- Registre I2C_CR1
  - Bit 8 – START : Permet de générer le START sur le bus
    - Les esclaves sont alors à l'écoute … pour la suite
  - Bit 9 – STOP : Permet de générer le STOP sur le bus
    - Les esclaves peuvent alors se rendormir éventuellement
  - Bit 10 - ACK: Permet de générer automatiquement un ACK sur le bus après l'envoi d'un byte (en mode écriture)
    - Si le bit est à 0 => Envoi d'un NACK

START

STOP

ACK

NACK

26.6.1 I²C Control register 1 (I2C_CR1)

Address offset: 0x00
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRST	Res.	ALERT	PEC	POS	ACK	STOP	START	NO STRETCH	ENGC	ENPEC	ENARP	SMB TYPE	Res.	SMBUS	PE
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw

Registres de contrôle I2C STM32

- Registre I2C_CR2
  - Bit 5 à 0 – FREQ[5:0] : Permet de configurer la clock interne du périphérique I2C (*Attention on ne parle pas de la clock SCL ici* !!)
  - Par contre, cette Horloge servira de base au calcul pour la fréquence SCL

Bits 5:0 **FREQ[5:0]**: Peripheral clock frequency

The FREQ bits must be configured with the APB clock frequency value (I2C peripheral connected to APB). The FREQ field is used by the peripheral to generate data setup and hold times compliant with the I2C specifications. The minimum allowed frequency is 2 MHz, the maximum frequency is limited by the maximum APB1 frequency and cannot exceed 50 MHz (peripheral intrinsic maximum limit).

0b000000: Not allowed
0b000001: Not allowed
0b000010: 2 MHz
...
0b110010: 50 MHz
Higher than 0b100100: Not allowed

26.6.2 I²C Control register 2 (I2C_CR2)

Address offset: 0x04
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			LAST	DMA EN	ITBUF EN	ITEVTE N	ITERR EN	Reserved		FREQ[5:0]					
			rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw

Registres de Contrôle I2C STM32

- Registre I2C_CCR
  - Permet de configurer la vitesse de communication (la fréquence sur SCL)
    - Voir formule de la période Thigh/Tlow dans le mode Slow Mode.

26.6.8 I²C Clock control register (I2C_CCR)

Address offset: 0x1C
Reset value: 0x0000

Note: *f_{PCLK1} must be at least 2 MHz to achieve Sm mode I²C frequencies. It must be at least 4 MHz to achieve Fm mode I²C frequencies. It must be a multiple of 10MHz to reach the 400 kHz maximum I²C Fm mode clock.*

The CCR register must be configured only when the I2C is disabled (PE = 0).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	Reserved		CCR[11:0]											
rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15 **F/S:** I2C master mode selection
0: Sm mode I2C
1: Fm mode I2C

Bits 11:0 **CCR[11:0]:** Clock control register in Fm/Sm mode (Master mode)
Controls the SCL clock in master mode.
Sm mode or SMBus:
$T_{high} = CCR * T_{PCLK1}$
$T_{low} = CCR * T_{PCLK1}$

Registres de Contrôle I2C STM32

- Registre I2C_TRISE
  - Permet de configurer le temps de montée pour respecter la norme I2C
    - En mode SM utiliser le formule donnée dans la datasheet :
      - *TRISE = (1000 / Tpclk1) + 1*
      - *Tpclk1 est la période donnée par le registre I2C_CCR*

26.6.9 I²C TRISE register (I2C_TRISE)

Address offset: 0x20
Reset value: 0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										TRISE[5:0]					
										rw	rw	rw	rw	rw	rw

Bits 5:0 **TRISE[5:0]**: Maximum rise time in Fm/Sm mode (Master mode)
These bits should provide the maximum duration of the SCL feedback loop in master mode.
The purpose is to keep a stable SCL frequency whatever the SCL rising edge duration.
These bits must be programmed with the maximum SCL rise time given in the I²C bus specification, incremented by 1.
For instance: in Sm mode, the maximum allowed SCL rise time is 1000 ns.
If, in the I2C_CR2 register, the value of FREQ[5:0] bits is equal to 0x08 and T_{PCLK1} = 125 ns therefore the TRISE[5:0] bits must be programmed with 09h.
(1000 ns / 125 ns = 8 + 1)

Registres de Contrôle I2C STM32

- Registre I2C_DR
  - Permet d'envoyer une donnée 8 bits sur le bus I2C
    - Utilisé pour envoyer l'adresse
    - Et des données au SLAVE
  - Permet de lire une donnée 8 bits sur le bus I2C
    - Permet de recevoir des données du SLAVE

26.6.5 I²C Data register (I2C_DR)

Address offset: 0x10
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DR[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

Registres de Statut I2C STM32

- Registre de STATUT I2C_SR1
  - Bit SB va passer à 1 quand le START est effectivement placé sur la ligne I2C
  - Bit ADDR va passer à 1 quand l'ADDResse est effectivement placé sur la ligne I2C
  - Bit TxE (utile en mode transmission!)
    - Va passer à 1 quand le registre DR est vide et que l'on peut donc transmettre une autre DATA
  - Bit RxNE (utile en mode réception!)
    - Va passer à 1 quand une DATA a été reçue dans le registre DR
  - *Signification : RC_W0 = Read & Clear ou Write at 0 Only !*
    - *La lecture du bit, efface ce bit !*
  - *Les Bit TxE et RxNE sont effacés sur écriture/lecture du registre DR*

26.6.6 I²C Status register 1 (I2C_SR1)

Address offset: 0x14
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMB ALERT	TIME OUT	Res.	PEC ERR	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTF	ADDR	SB
rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	r	r		r	r	r	r	r

Registres de Statut I2C STM32

- Registre de STATUS I2C_SR2
  - EN mode maitre, il servira simplement à effacer le bit de status ADDR du I2C_SR1 en le lisant.

26.6.7 I²C Status register 2 (I2C_SR2)

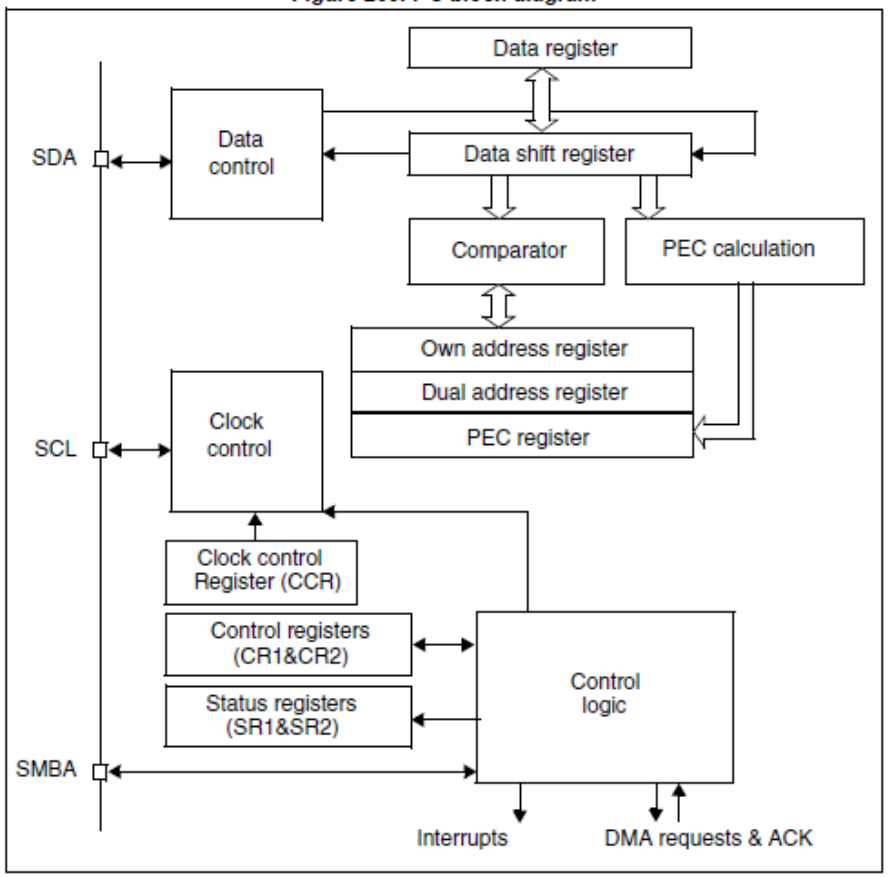
Address offset: 0x18
Reset value: 0x0000

Note: Reading I2C_SR2 after reading I2C_SR1 clears the ADDR flag, even if the ADDR flag was set after reading I2C_SR1. Consequently, I2C_SR2 must be read only when ADDR is found set in I2C_SR1 or when the STOPF bit is cleared.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PEC[7:0]								DUALF	SMB HOST	SMBDE FAULT	GEN CALL	Res.	TRA	BUSY	MSL
r	r	r	r	r	r	r	r	r	r	r	r		r	r	r

Annexe 1: Vue Globale Macro I2C STM32LXXX

Figure 209. I²C block diagram



1. SMBA is an optional signal in SMBus mode. This signal is not applicable if SMBus is disabled.