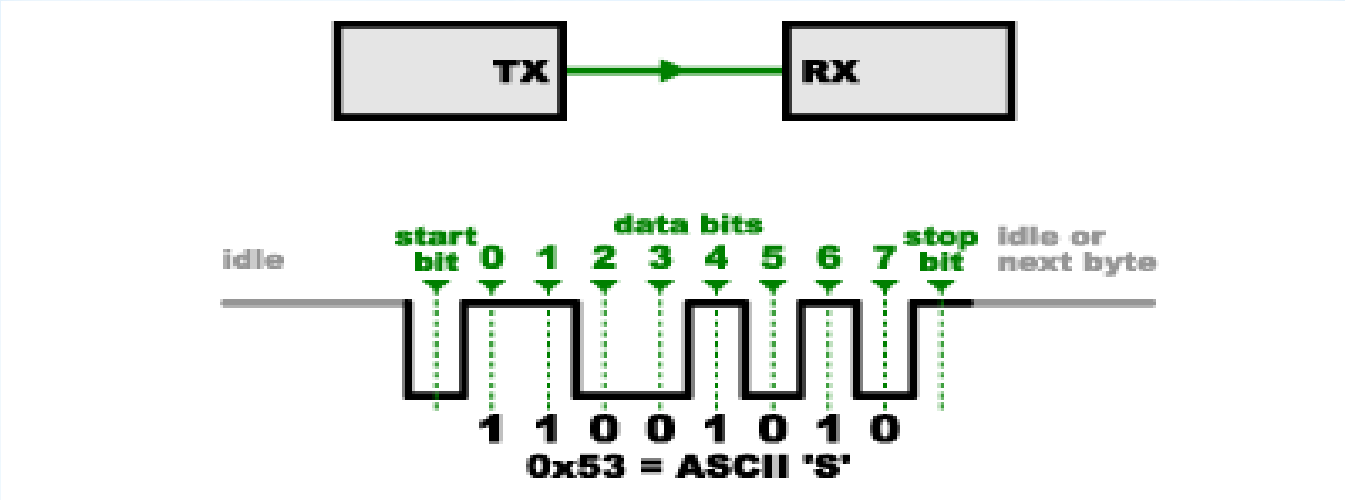
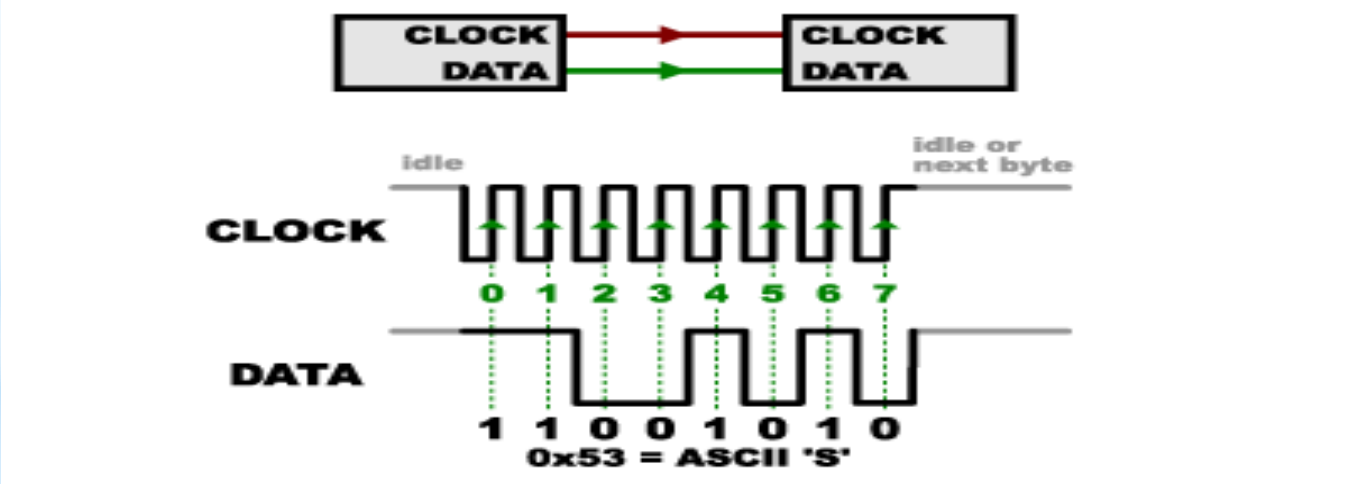




ΣΕΙΡΙΑΚΗ ΕΠΙΚΟΙΝΩΝΙΑ

Σύγχρονη σειριακή αποστολή δεδομένων

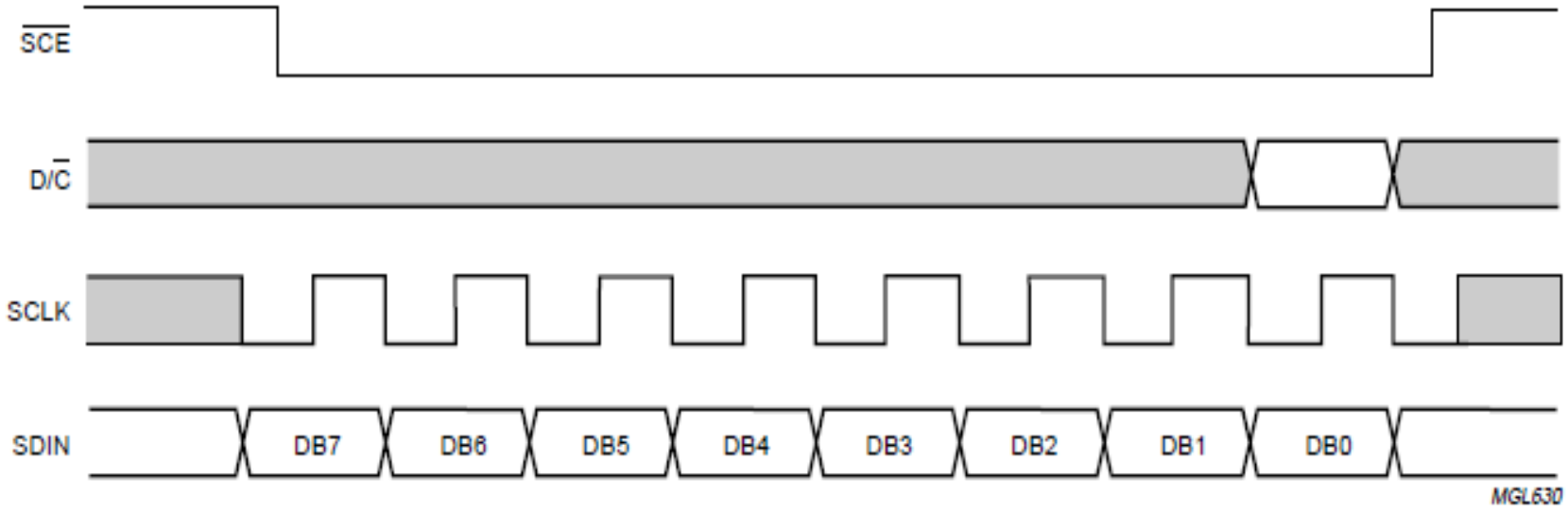
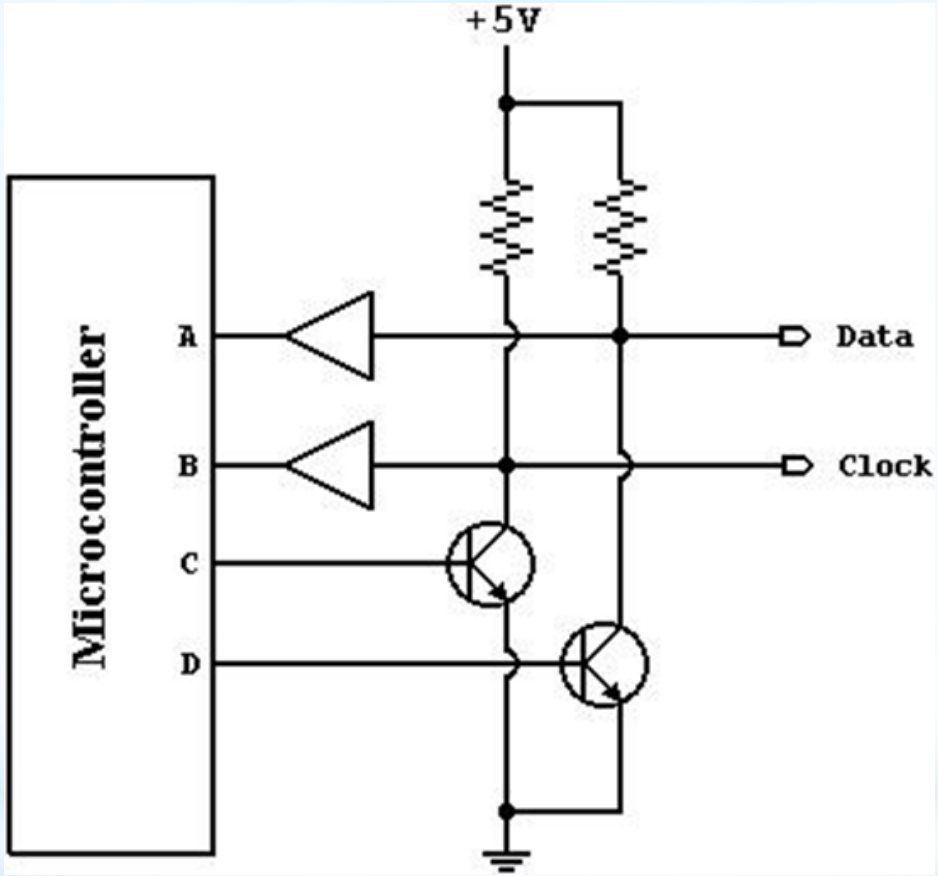


Fig.10 Serial bus protocol - transmission of one byte.

Σύνδεση PS/2 πληκτρολογίου και mouse



Γενική διεπαφή ανοικτού συλλέκτη. Τα δεδομένα και το ρολόι διαβάζονται στις ακίδες A και B του μικροελεγκτή, αντίστοιχα. Και οι δύο γραμμές συνήθως διατηρούνται στο +5V, αλλά μπορούν γειωθούν δίνοντας λογικό "1" σε C και D.

Γενική Περιγραφή Επικοινωνίας

Το ποντίκι PS/2 και το πληκτρολόγιο υλοποιούν ένα αμφίδρομο σύγχρονο σειριακό πρωτόκολλο. Το Bus είναι "αδρανές" όταν και οι δύο γραμμές είναι σε υψηλή τάση (open-collector).

Αυτή είναι η μόνη κατάσταση όπου επιτρέπεται το πληκτρολόγιο/ποντίκι να αρχίσει να μεταδίδει δεδομένα.

Ο μικρουπολογιστής έχει τον κύριο έλεγχο του bus και μπορεί να εμποδίσει την επικοινωνία ανά πάσα στιγμή τραβώντας τη γραμμή ρολογιού χαμηλά.

Η συσκευή παράγει πάντα το σήμα Clock. Εάν ο κεντρικός υπολογιστής θέλει να στείλει δεδομένα, πρέπει πρώτα να εμποδίσει την επικοινωνία από τη συσκευή τραβώντας το ρολόι χαμηλά.

Ο υπολογιστής γειώνει έπειτα τα data και απελευθερώνει το Clock. Αυτή είναι η κατάσταση "Αίτηση για αποστολή" και σηματοδοτεί τη συσκευή για να ξεκινήσει την παραγωγή παλμών ρολογιού.

Καταστάσεις του Διαύλου

Data = high, Clock = high: *Idle state*.
Data = high, Clock = low: *Communication Inhibited*.
Data = low, Clock = high: *Host Request-to-Send*

Όλα τα δεδομένα μεταδίδονται ένα byte κάθε φορά και κάθε byte αποστέλλεται σε ένα πλαίσιο που αποτελείται από 11-12 bit. Αυτά τα bits είναι:

1 start bit. Είναι πάντα 0.

8 data bits, Πρώτα το λιγότερο σημαντικό ψηφίο

1 parity bit (μονή ισοτιμία).

1 stop bit. Είναι πάντα 1.

1 acknowledge bit (μόνο σε επικοινωνία από host-to-device)

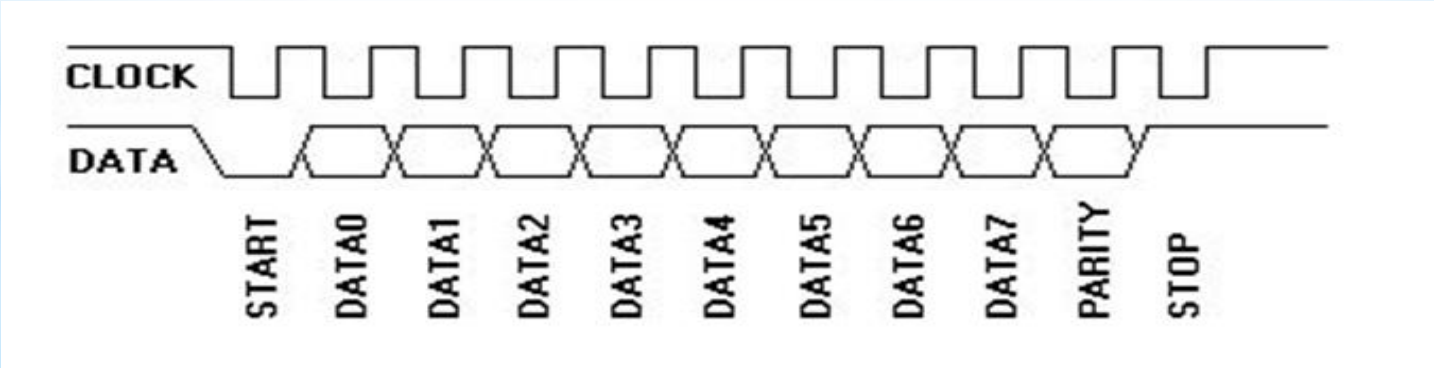
Χρονικές Προδιαγραφές

Τα δεδομένα που αποστέλλονται από τη συσκευή στον κεντρικό υπολογιστή διαβάζονται στο falling edge του Clock.

Τα δεδομένα που αποστέλλονται από τον κεντρικό υπολογιστή στη συσκευή διαβάζονται στο rising edge του Clock.

Το πληκτρολόγιο/ποντίκι γράφει ένα Bit στη γραμμή data όταν το Clock είναι High και το bit διαβάζεται από τον υπολογιστή όταν το Clock είναι Low.

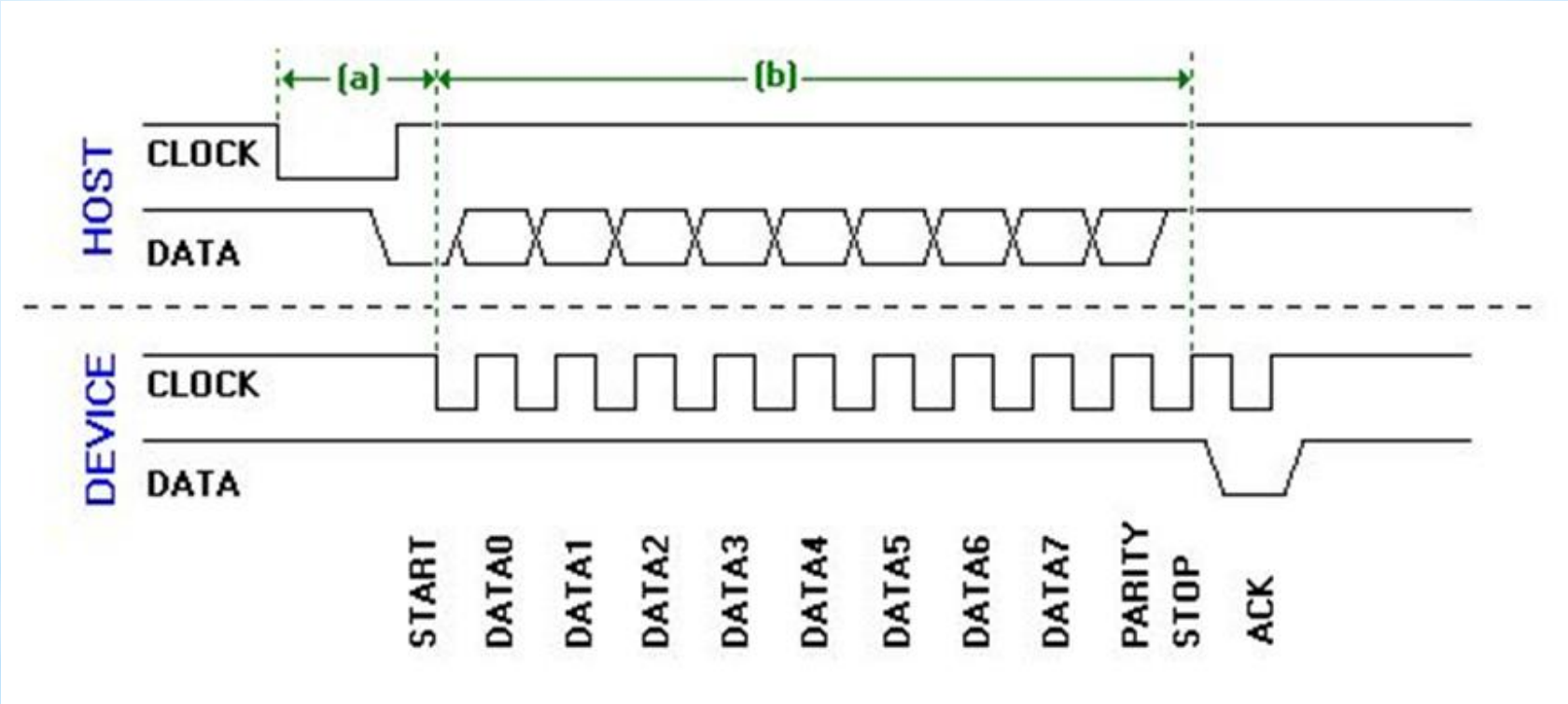
Η συχνότητα του Clock πρέπει να είναι 10 - 16,7 kHz. Αυτό σημαίνει ότι το Clock πρέπει να είναι υψηλό για 30 - 50 ms και χαμηλό για 30 – 50 ms



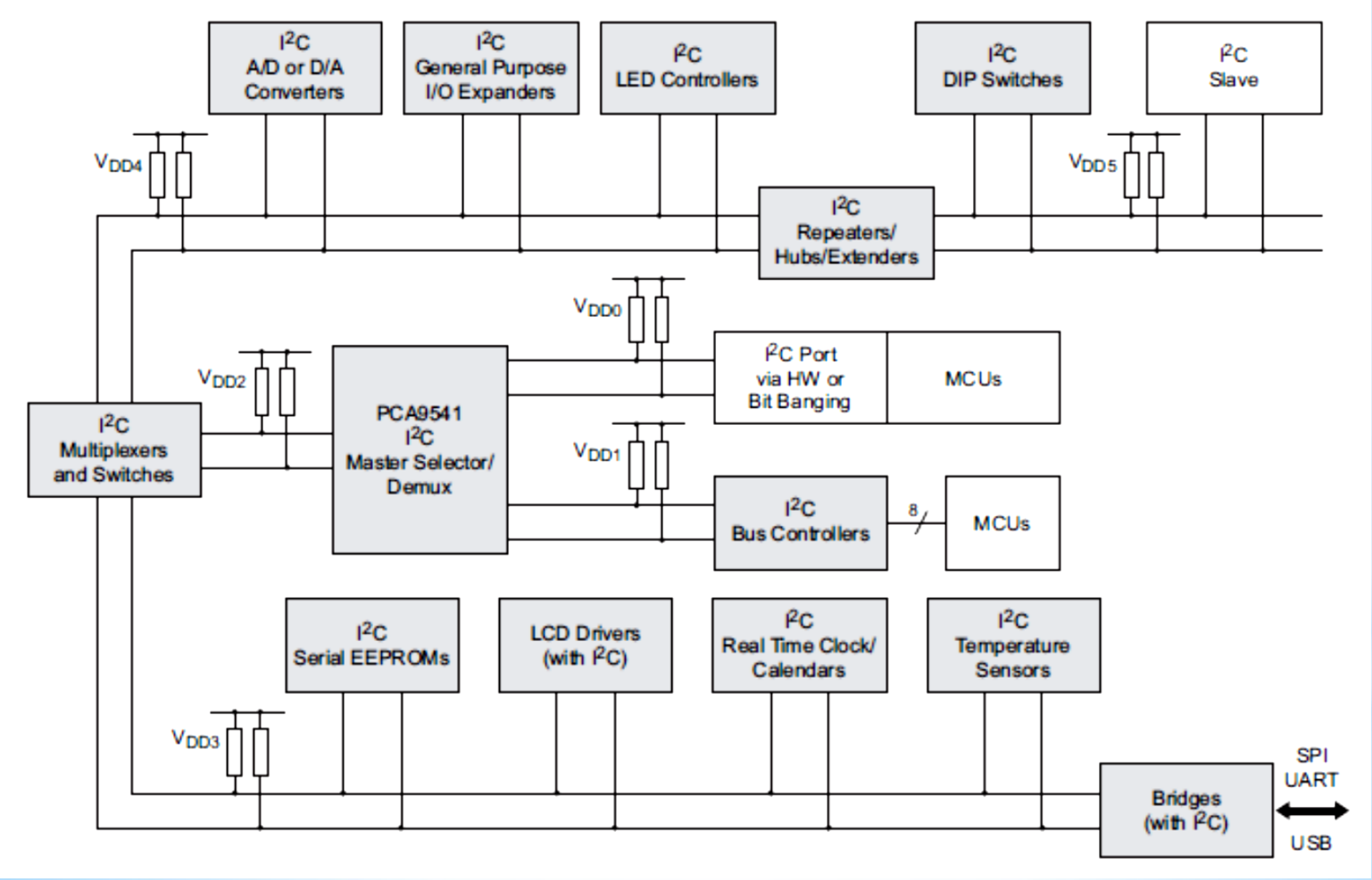
Παράδειγμα αποστολής χαρακτήρα

Ο κωδικός για το πλήκτρο "Q" key (15h) αποστέλλεται από το πληκτρολόγιο στον Υπολογιστή

START(0) 1 0 1 0 1 0 0 0 PAR STOP(1)


(a)
(b)
HOST
CLOCK
DATA
DEVICE
CLOCK
DATA
START
DATA0
DATA1
DATA2
DATA3
DATA4
DATA5
DATA6
DATA7
PARITY
STOP
ACK

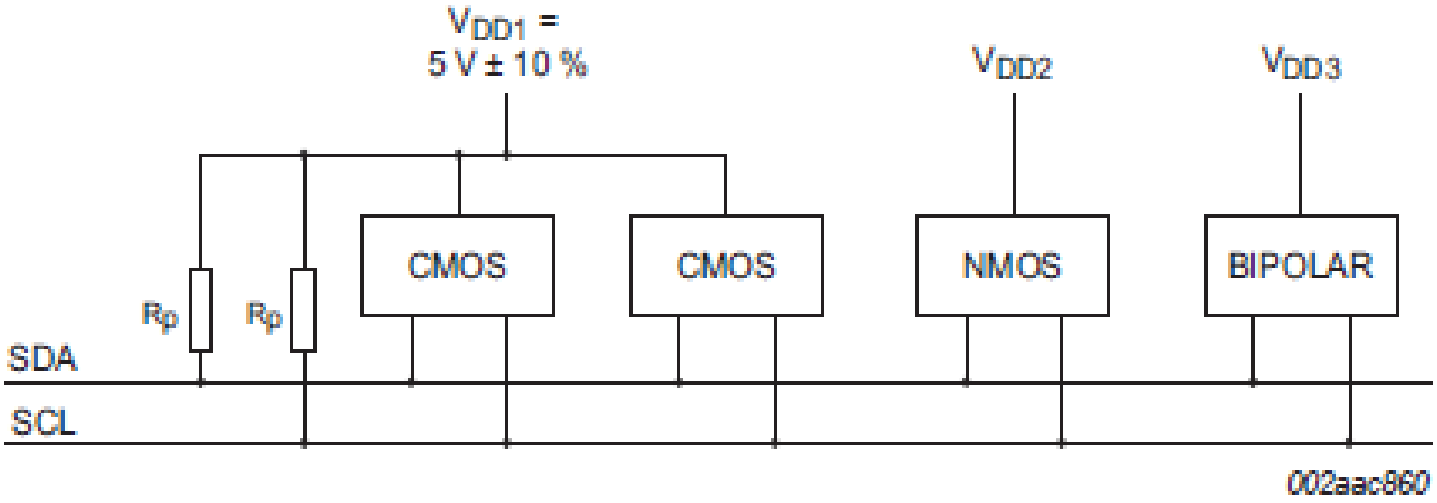
I2C COMMUNICATION PROTOCOL



I2C Ονοματολογία

Term	Description
Transmitter	Η συσκευή που στέλνει δεδομένα στον δίαυλο.
Receiver	Η συσκευή που δέχεται δεδομένα από τον δίαυλο.
Master	Η συσκευή που ξεκινάει μια μεταφορά, δημιουργεί τα σήματα του clock (SCL) και τερματίζει την μεταφορά
Slave	Η συσκευή στην οποία απευθύνεται ο master.
Multi-master	Περισσότεροι από ένας Master μπορούν να προσπαθήσουν να ελέγξουν το δίαυλο, χωρίς να καταστρέψουν το μήνυμα.
Arbitration	Διαδικασία που διασφαλίζει ότι, αν περισσότεροι από ένας master προσπαθήσει να ελέγξει τον δίαυλο, μόνο ένας μπορεί να επικρατήσει και το μήνυμα που στέλνει δεν αλλοιώνεται.
Synchronization	Διαδικασία για τον συγχρονισμό της ταχύτητας δύο συσκευών.

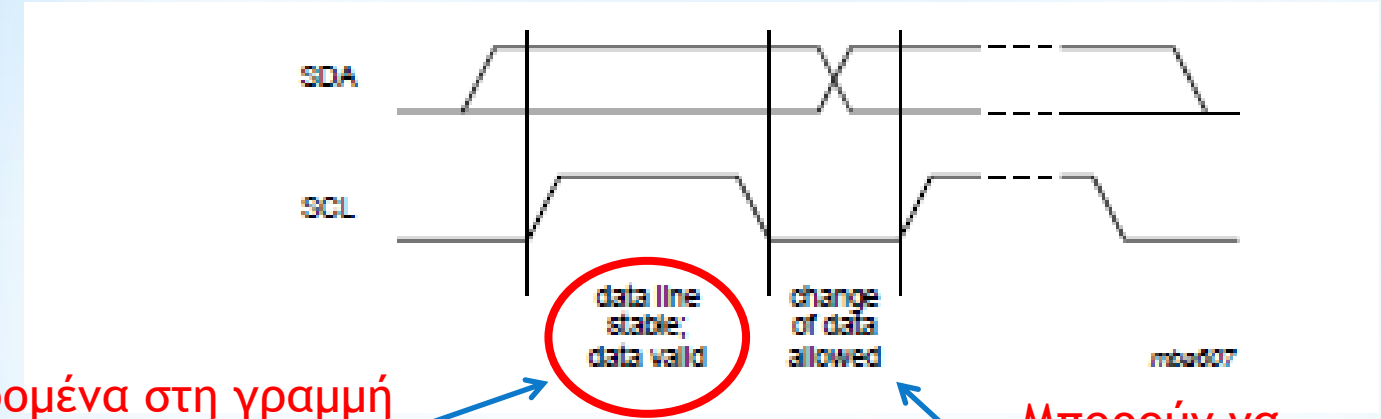
I2C Σύνδεση συσκευών με διαφορετικές τάσεις τροφοδοσίας



Οι συνδέσεις κάθε συσκευής με τον δίαυλο πρέπει να είναι ανοιχτού συλλέκτη για να μπορεί να πραγματοποιηθεί η σχέση wired –AND.

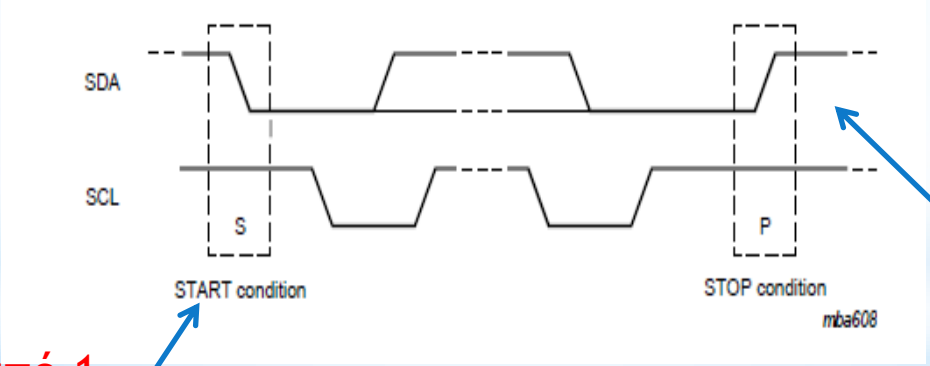
Όταν μια συσκευή προσπαθεί να γράψει 0 και μια άλλη 1 τότε μια από τις δύο πρέπει να απορροφήσει αρκετά μεγάλη ένταση.

I2C Προδιαγραφές σημάτων SCL και SDA



Τα δεδομένα στη γραμμή SDA πρέπει να είναι αμετάβλητα όταν SCL είναι HIGH.

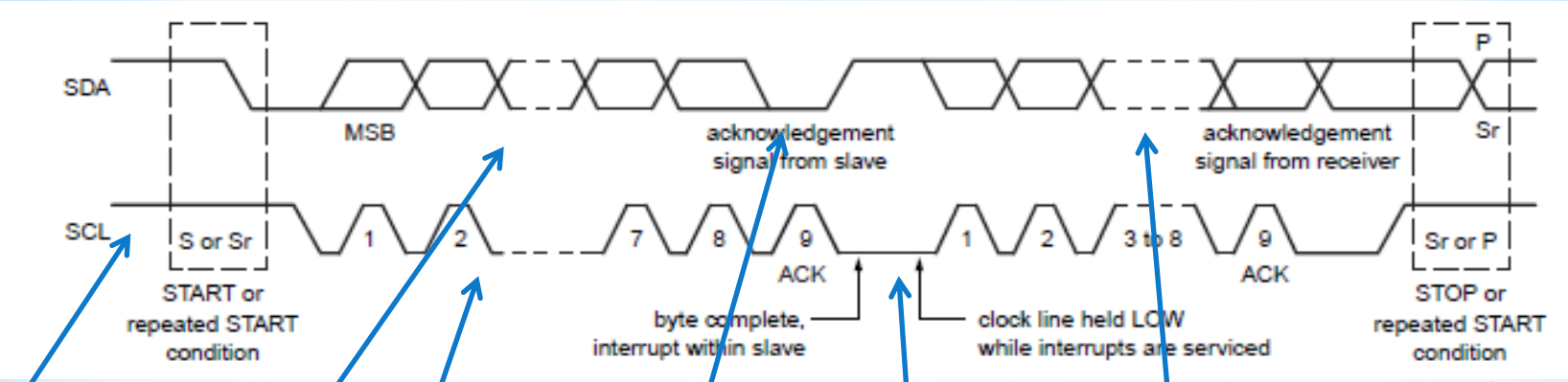
Μπορούν να αλλάξουν μόνο όταν SCL είναι LOW.



Αλλαγή στο SDA από 1 σε 0 όταν SCL είναι 1 δηλώνει αρχή μετάδοσης (START)

Αλλαγή στο SDA από 0 σε 1 όταν SCL είναι 1 δηλώνει τέλος μετάδοσης (STOP)

Μεταφορά δεδομένων σε I2C δίαυλο



Αρχή Μεταφοράς

Διεύθυνση του Slave

Το Clock προέρχεται από τον Master που ξεκινά την μεταφορά.

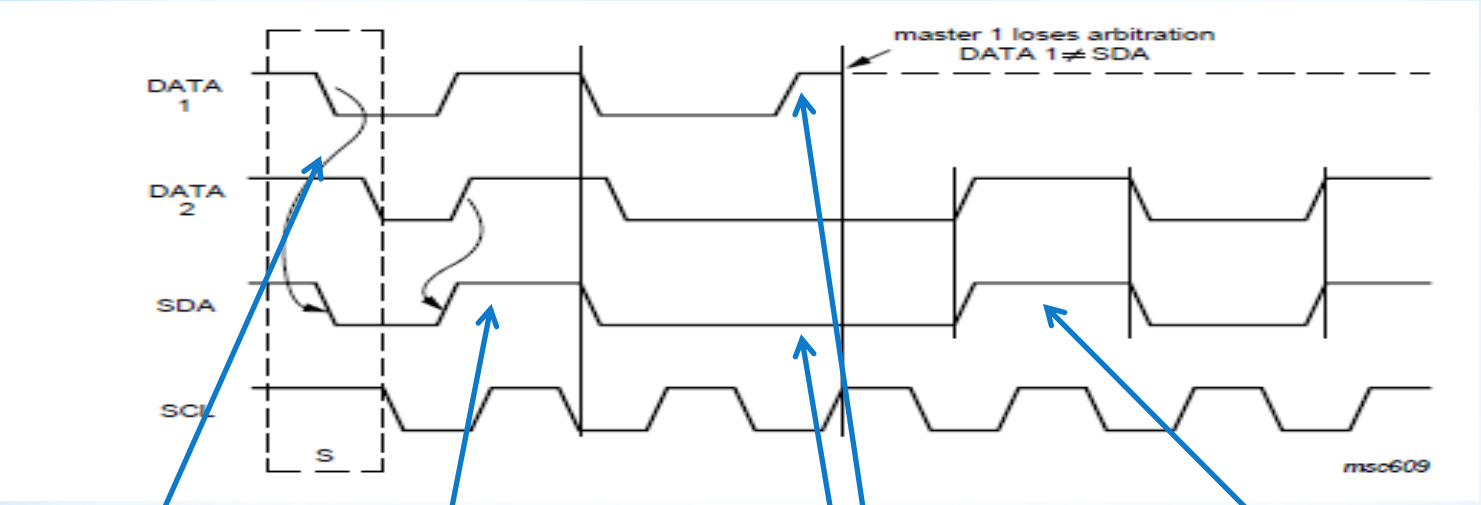
Στο τελος του byte ο slave στέλνει ένα ACK.

Ο slave μπορεί να κρατάει το SCL LOW για να αναγκάσει τον master να περιμένει.

Μεταφορά από master ή από slave (R/W) και ACK από παραλήπτη

Τέλος μεταφοράς

I2C Arbitration



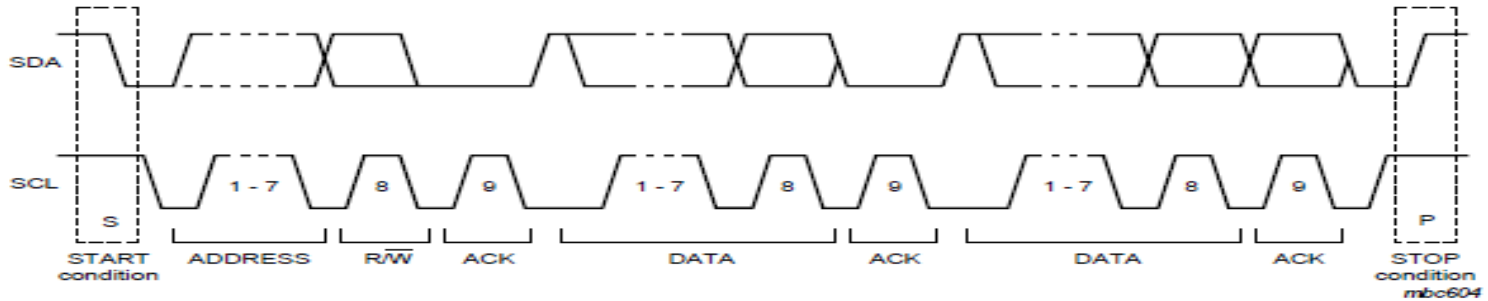
Δύο Masters στέλνουν ταυτόχρονα δεδομένα στο δίαυλο (DATA1 και DATA2).

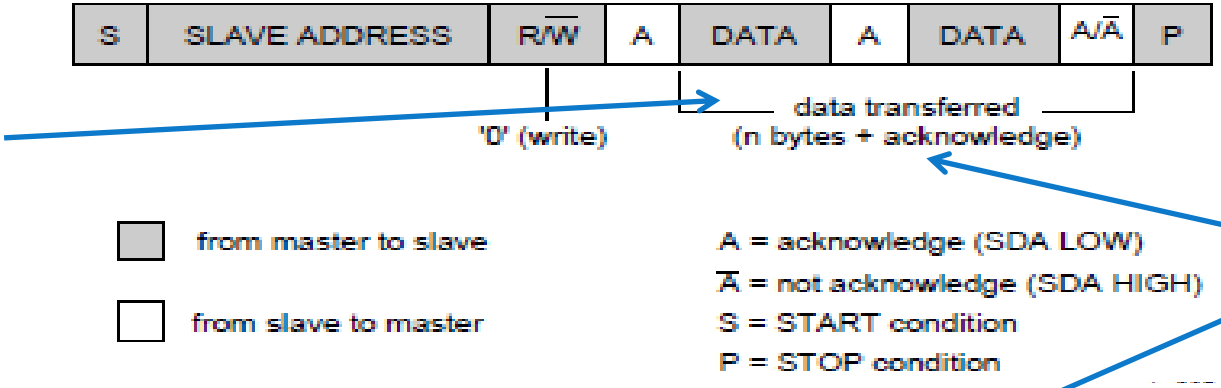
Και οι δυο Masters συγκρίνουν αυτό που έστειλαν με την τιμή του SDA.

Ο πρώτος που θα διαπιστώσει διαφορά σταματά την χρήση του διαύλου.

Ο άλλος συνεχίζει κανονικά την μεταφορά δεδομένων.

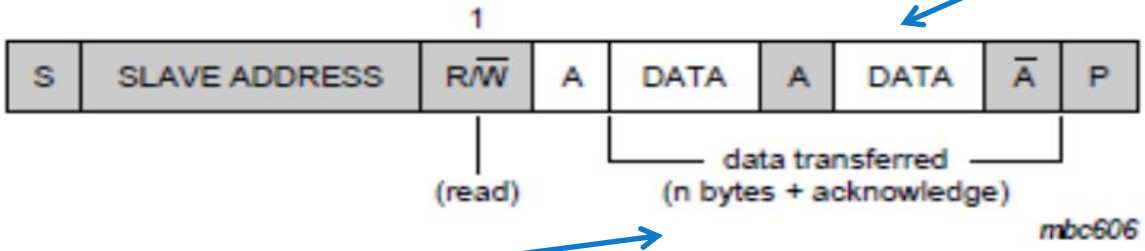
I2C Παράδειγμα μεταφοράς από master σε slave





Data από master, ACK από slave

Πολλαπλά Data στο ίδιο frame



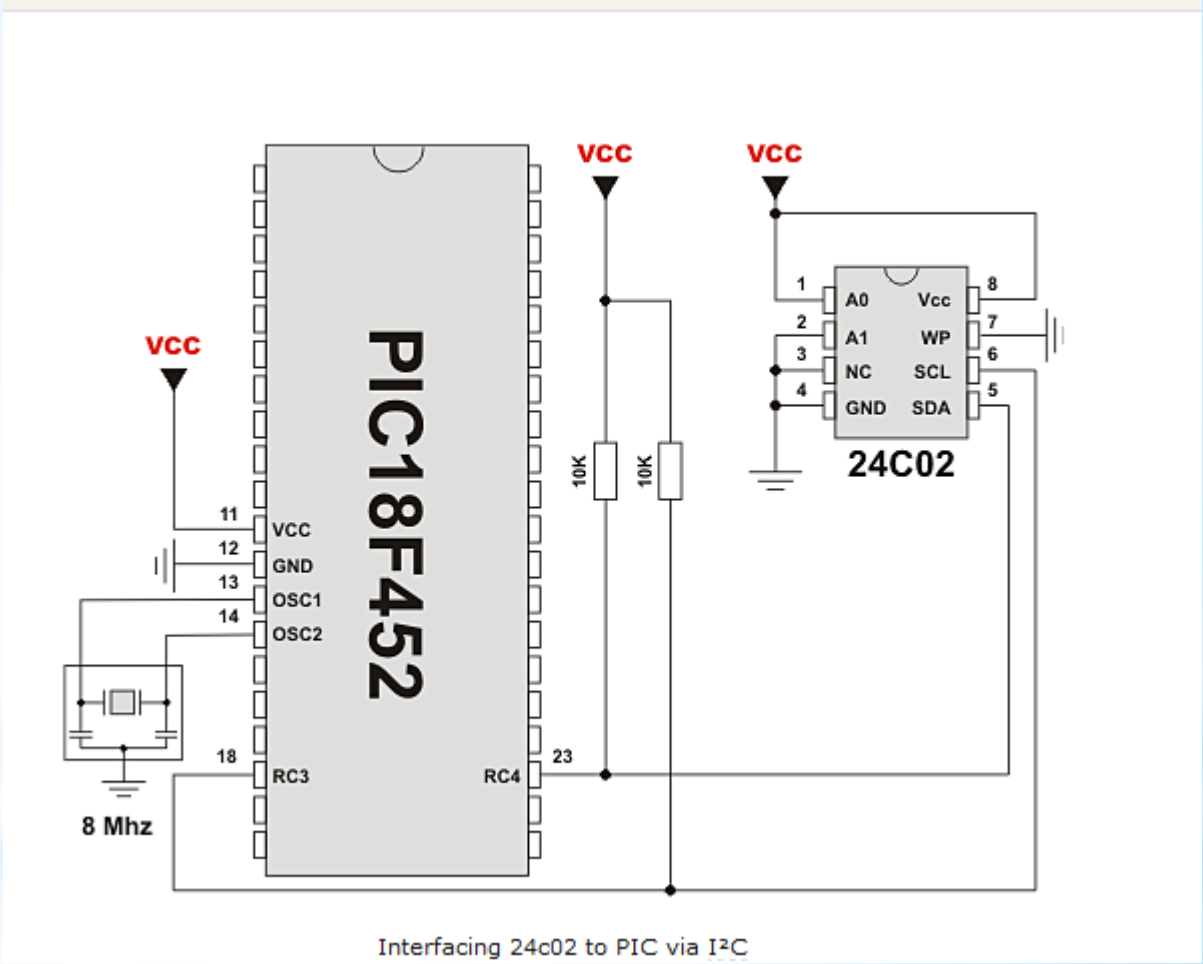
Data από master, ACK από slave

Παράδειγμα εφαρμογής I2C

Σύνδεση σειριακής EEPROM 24C02

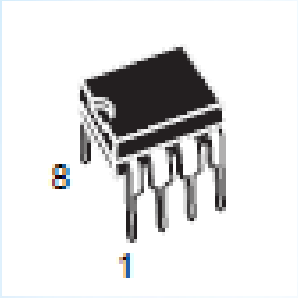
με μικροελεγκτή

μέσω I2C.



Interfacing 24c02 to PIC via I²C

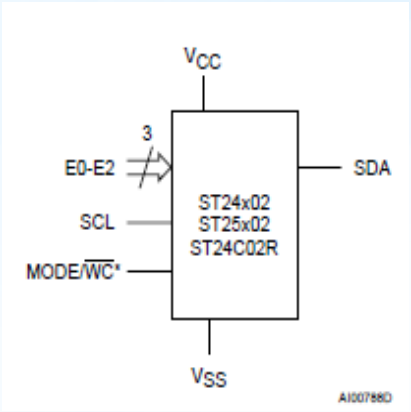
Σειριακή I2C EEPROM 24C02



256X8 EEPROM

Συμβατή με το πρωτόκολλο I2C

Δυνατότητα Εγγραφής και ανάγνωσης ενός ή πολλαπλών byte.



Λογικό Διάγραμμα

E0-E2	Chip Enable Inputs
SDA	Serial Data Address Input/Output
SCL	Serial Clock
MODE	Multibyte/Page Write Mode (C version)
$\overline{WC}$	Write Control (W version)
V_{CC}	Supply Voltage
V_{SS}	Ground

Πίνακας Σημάτων

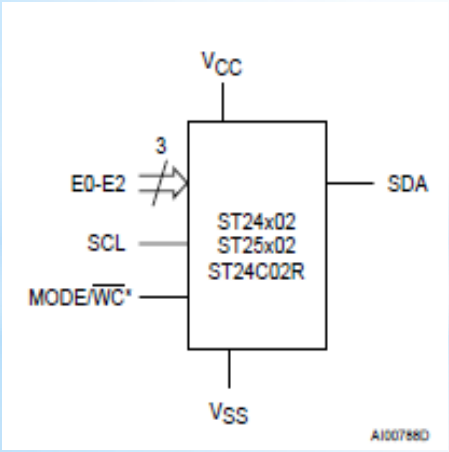
- TWO WIRE SERIAL INTERFACE, FULLY I^2C BUS COMPATIBLE
- BYTE and MULTIBYTE WRITE (up to 4 BYTES)
- PAGE WRITE (up to 8 BYTES)
- BYTE, RANDOM and SEQUENTIAL READ MODES
- SELF TIMED PROGRAMMING CYCLE
- AUTOMATIC ADDRESS INCREMENTING

Βασικά χαρακτηριστικά

Εγγραφή δεδομένων στην 24C02 EEPROM

Η I2C διεύθυνση της EEPROM είναι εργοστασιακά 1010.

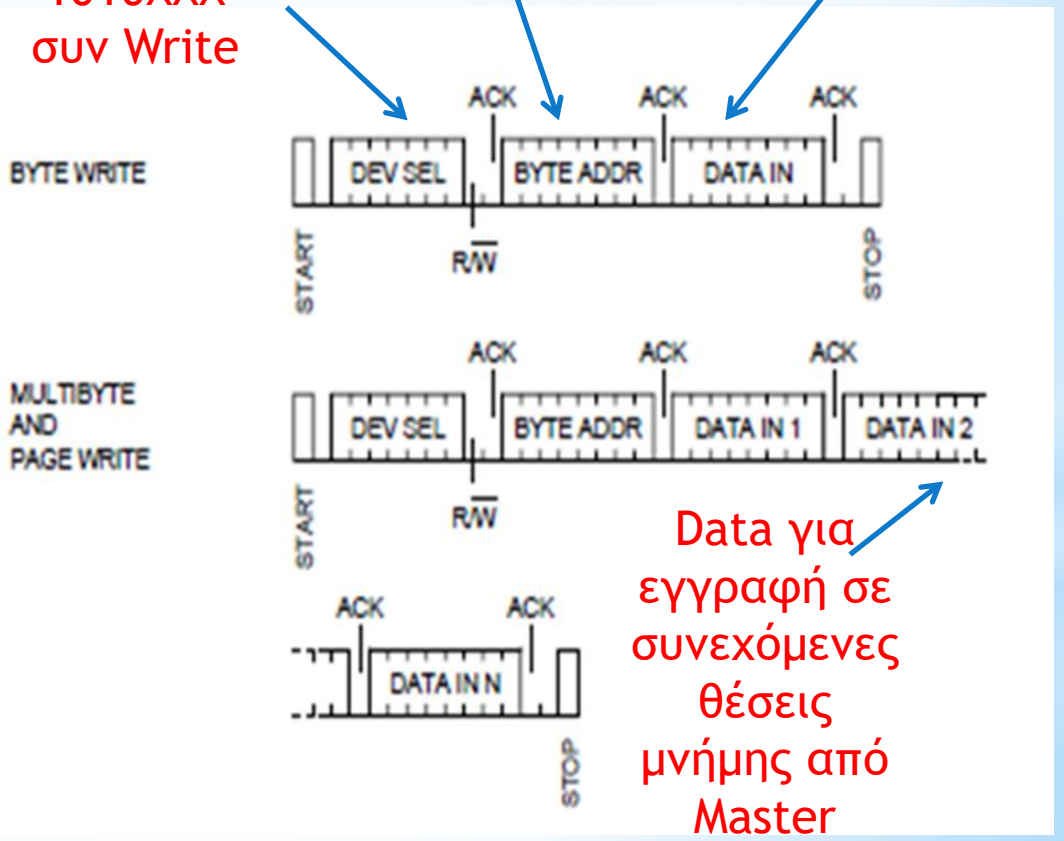
Επιπλέον οι είσοδοι E2 E1 E0 μπορούν να χρησιμοποιηθούν για σύνδεση μέχρι 8 συσκευών στον ίδιο δίαυλο.



Διεύθυνση συσκευής (από master) 1010XXX συν Write

Θέση Μνήμης για εγγραφή (0-255) Από Master

Data για εγγραφή από Master



Data για εγγραφή σε συνεχόμενες θέσεις μνήμης από Master

Σειρά εντολών I2C για εγγραφή ενός byte σε EEPROM

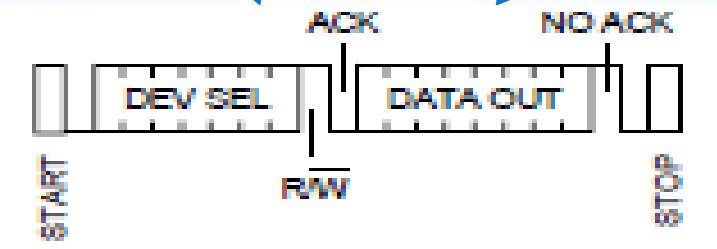
* Ελέγχουμε αν ο δίαυλος είναι αδρανής (SDA, SCL high)
* Στέλνουμε ένα I2C Start (SCL High ακολουθούμενο από SDA low)
* Στέλνουμε ένα byte με την διεύθυνση της EERPOM και 0 για Write (1010XXX0)
* Στέλνουμε ένα byte με την θέση μνήμης στην οποία θέλουμε να εγγραφούν τα δεδομένα (0-255)
* Στέλνουμε ένα byte με τα δεδομένα
* Στέλνουμε ένα Stop (SCL high ακολουθούμενο από SDA high)

Ανάγνωση δεδομένων από 24C02 EEPROM

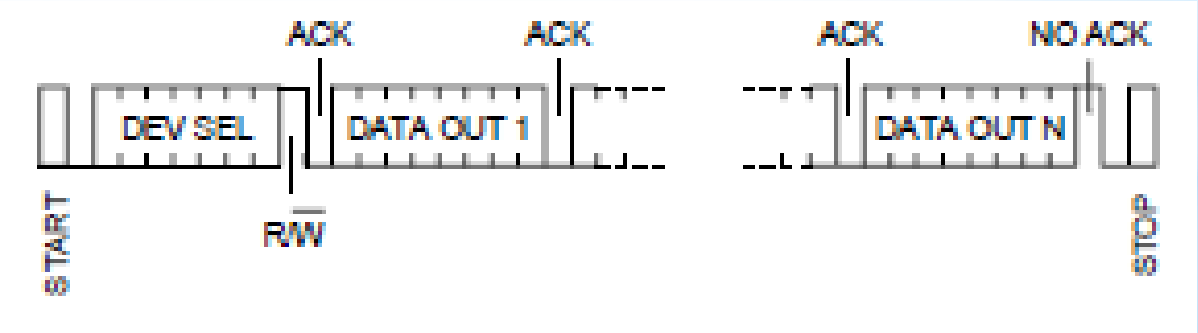
Διεύθυνση συσκευής (από master) 1010XXX συν Read

Data από EEPROM

Ανάγνωση από τρέχουσα θέση μνήμης.



Συνεχόμενη ανάγνωση από τρέχουσα θέση μνήμης



Ανάγνωση δεδομένων από 24C02 EEPROM

Διεύθυνση συσκευής (από master) 1010XXX συν Write

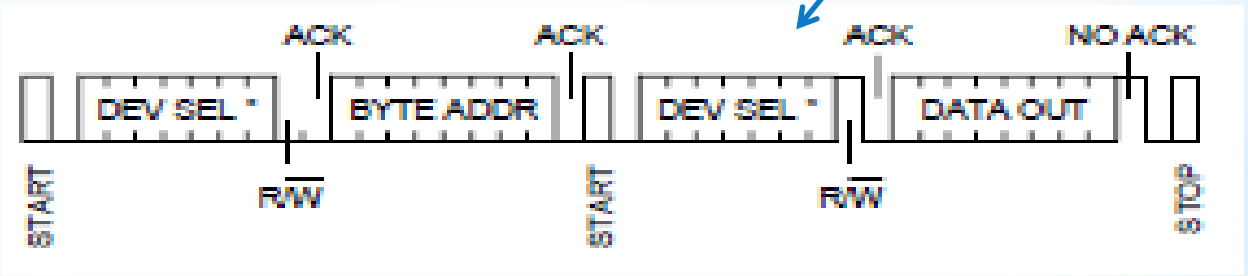
Θέση Μνήμης για ανάγνωση (0-255) Από Master

ReStart

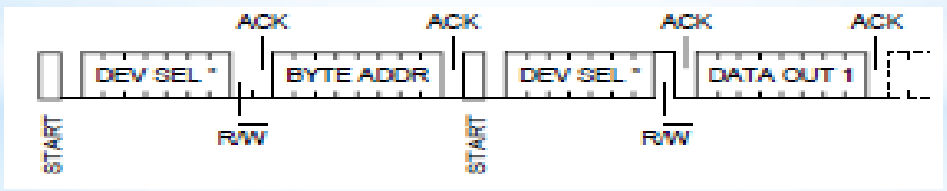
Διεύθυνση συσκευής (από master) 1010XXX συν Read

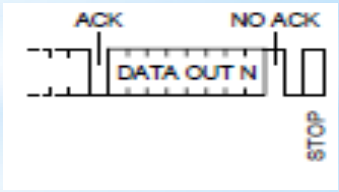
Data από EEPROM

Ανάγνωση από οριζόμενη θέση μνήμης.



Συνεχόμενη ανάγνωση από οριζόμενη θέση μνήμης





Σειρά εντολών I2C για ανάγνωση ενός byte από EEPROM

* Ελέγχουμε αν ο δίαυλος είναι αδρανής (SDA, SCL high)
* Στέλνουμε ένα I2C Start (SCL High ακολουθούμενο από SDA low)
* Στέλνουμε ένα byte με την διεύθυνση της EERPOM και 0 για Write (1010XXX0)
* Στέλνουμε ένα byte με την θέση μνήμης στην οποία θέλουμε να εγγραφούν τα δεδομένα (0-255)
* Στέλνουμε ξανά ένα I2C Start (Restart)
* Στέλνουμε ένα byte με την διεύθυνση της EERPOM και 1 για Read (1010XXX1)
* Διαβάζουμε ένα byte με τα δεδομένα
* Στέλνουμε ένα Stop (SCL high ακολουθούμενο από SDA high)

Παράδειγμα κώδικα για εγγραφή και ανάγνωση από EEPROM

```
void main(){

  I2C_Init(100000);       // Αρχικοποίηση συστήματος I2C του μικροελεγκτή

  I2C_Start();            // Στέλνουμε I2C Start
  I2C_Wr(0xA2);           // Στέλνουμε την διεύθυνση της EEPROM + W
  I2C_Wr(2);              // Στέλνουμε την θέση μνήμης της EEPROM που θέλουμε να γράψουμε
  I2C_Wr(0xF0);           // Στέλνουμε δεδομένα  (Εγγραφή στην EEPROM)
  I2C_Stop();             // Στέλνουμε I2C Stop

  I2C_Start();              //  Στέλνουμε I2C start signal
  I2C_Wr(0xA2);             // Στέλνουμε διεύθυνση της EEPROM + W
  I2C_Wr(2);              // Στέλνουμε την θέση μνήμης της EEPROM που θέλουμε να διαβάσουμε
  I2C_Repeated_Start();   // Στέλνουμε I2C signal repeated start
  I2C_Wr(0xA3);             // Στέλνουμε διεύθυνση της EEPROM  + R
  data  = I2C_Rd(0);       // Διαβάζουμε τα δεδομένα από EEPROM (NO acknowledge)
  I2C_Stop();
}
```

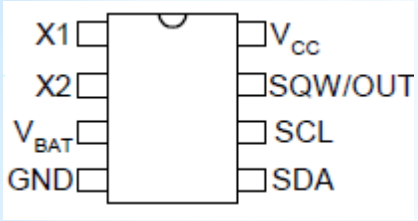
Παράδειγμα εγγραφής σε I2C EEPROM πολλαπλών bytes

```
void main(){

   Soft_I2C_Config(&PORTC, 4, 3); //Use PortC pins 4 and 3

   Soft_I2C_Start();              // Issue I2C start signal
   Soft_I2C_Write(0xA2);          // Send byte via I2C  (Address of 24c02)
  Soft_I2C_Write(2);              // Send byte (address of EEPROM location)
  Soft_I2C_Write('A');            // Send data (data to be written)
  Soft_I2C_Write('B');            // Send data (data to be written)
  Soft_I2C_Write('C');            // Send data (data to be written)
  Soft_I2C_Write('1');            // Send data (data to be written)
  Soft_I2C_Write('2');            // Send data (data to be written)

  Soft_I2C_Stop();
}
```

Παράδειγμα ανάγνωσης από I2C EEPROM πολλαπλών bytes

```
void main(){
  char mem_data[10];

  Soft_I2C_Config(&PORTC, 4, 3); //Use Port C pins 4 and 3
  Soft_I2C_Start();              // Issue I2C start signal
  Soft_I2C_Write(0xA2);          // Send byte (device address + W)
  Soft_I2C_Write(8);             // Send byte (EEPROM location to read from)

  Soft_I2C_Start();              // Issue I2C start signal
  Soft_I2C_Write(0xA3);          // Send byte (device address + R)
  mem_data[0] = Soft_I2C_Read(1);    // Read data (send ACK)
  mem_data[1] = Soft_I2C_Read(1);    // Read data (send ACK)
  mem_data[2] = Soft_I2C_Read(1);    // Read data (send ACK)
  mem_data[3] = Soft_I2C_Read(0);    // Read data (NO ACK)

Soft_I2C_Stop();
}
```

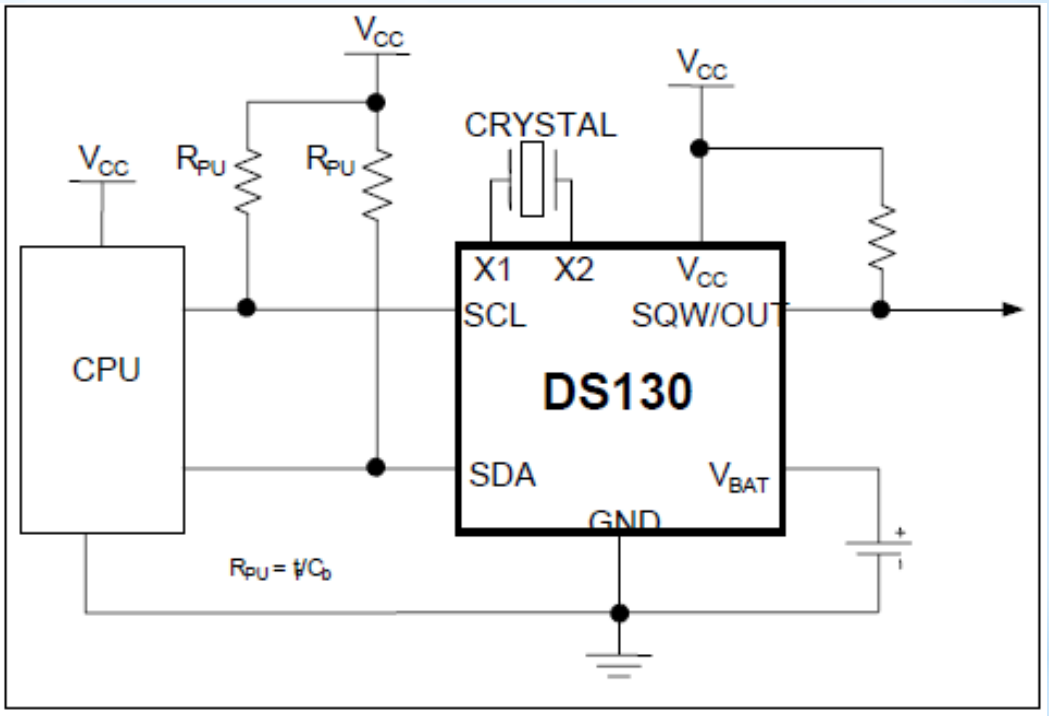
I2C Real-Time Clock DS1307



Συμβατό με το πρωτόκολλο I2C

Η I2C διεύθυνση είναι 1101000? (0xD0, 0XD1).

Δυνατότητα Εγγραφής και ανάγνωσης ενός ή πολλαπλών byte.



Τυπική σύνδεση

I2C Real-Time Clock DS1307

ADDRESS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNCTION	RANGE
00h	CH	10 Seconds			Seconds				Seconds	00–59
01h	0	10 Minutes			Minutes				Minutes	00–59
02h	0	12	10 Hour	10 Hour	Hours				Hours	1–12 +AM/PM 00–23
		24	PM/ AM							
03h	0	0	0	0	0	DAY			Day	01–07
04h	0	0	10 Date		Date				Date	01–31
05h	0	0	0	10 Month	Month				Month	01–12
06h	10 Year				Year				Year	00–99
07h	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08h–3Fh									RAM 56 x 8	00h–FFh

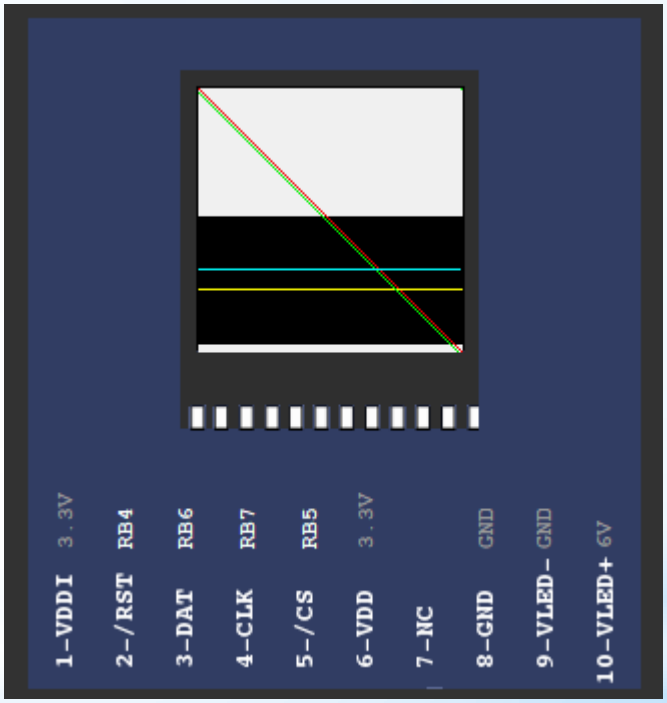
Εσωτερικοί Καταχωρητές του DS1307

Παράδειγμα ανάγνωσης λεπτών από TRC DS1307

```
void main(){
  unsigned char minutes[2];
  unsigned char temp;

  Soft_I2C_Config(&PORTC, 4, 3); //Use Port C pins 4 and 3
  Soft_I2C_Start();              // Issue I2C start signal
  Soft_I2C_Write(0xD0);          // Send byte (device address + W)
  Soft_I2C_Write(1);             // Send byte (Location of minutes register)

  Soft_I2C_Start();              // Issue I2C start signal
  Soft_I2C_Write(0xD1);          // Send byte (device address + R)
  temp = Soft_I2C_Read(0);       // Read data (NO ACK)
  Soft_I2C_Stop();

  minutes[0]=((temp  & 0x70)>>4)+0x30;
  minutes[1]= (temp  & 0x0F) +0x30;
}
```

Graphics LCD PCF8833 με σειριακό interface

* 132X132 pixels
* Κάθε pixel μπορεί να έχει μέχρι 4K διαφορετικά χρώματα.
* Σύγχρονη Σειριακή Επικοινωνία
* Παράλληλη Επικοινωνία



Σειριακή επικοινωνία

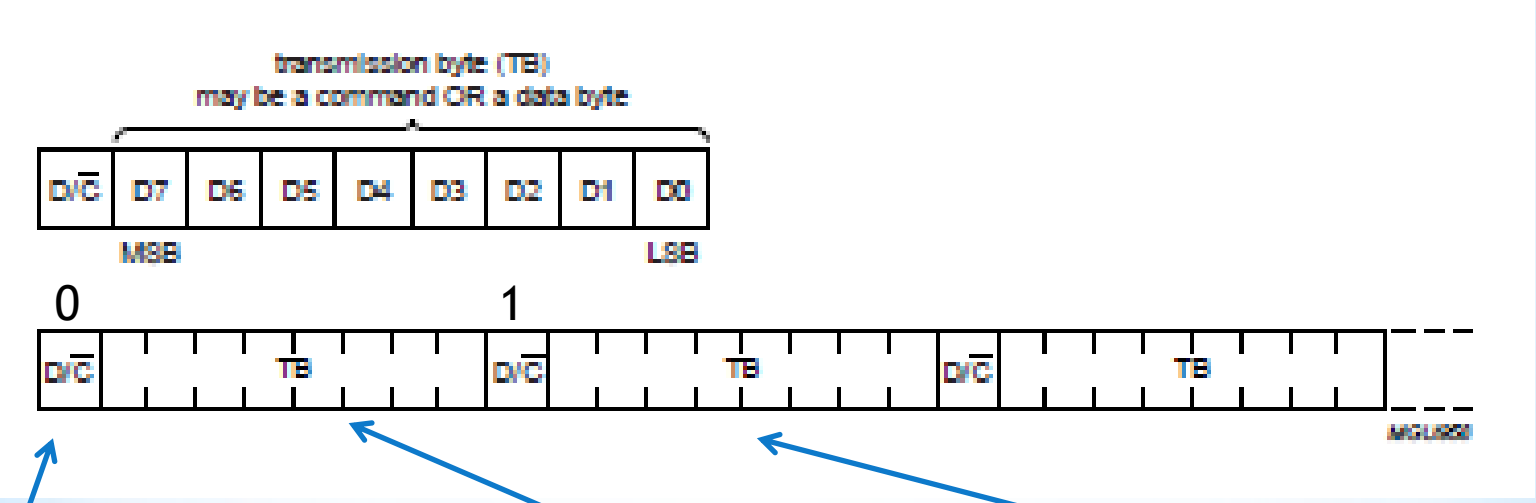
/RESET

DAT : Σειρακά δεδομένα

CLK : Clock

/CS : Chip Select

Graphics LCD PCF8833: Format σειριακής επικοινωνίας



Πριν από κάθε byte στέλνουμε ένα bit που καθορίζει εντολή (0) ή δεδομένα (1)

Αν D/C bit είναι 0 τότε D7..D0 αντιστοιχούν στον κωδικό μιας εντολής

Αν D/C bit είναι 1 τότε D7..D0 αντιστοιχούν σε δεδομένα

Graphics LCD PCF8833: Πίνακας Εντολών interface

D/C̄	7	6	5	4	3	2	1	0	DEFAULT	OTP	DESCRIPTION
0	0	0	0	0	0	0	0	0	00H	–	no operation (NOP)
0	0	0	0	0	0	0	0	1	01H	–	software reset (SWRESET)
0	0	0	0	0	0	0	1	0	02H	–	booster voltage off (BSTROFF)
0	0	0	0	0	0	0	1	1	03H	–	booster voltage on (BSTRON)
0	0	0	0	1	0	0	0	0	10H	–	Sleep IN
0	0	0	0	1	0	0	0	1	11H	–	Sleep_OUT
0	0	0	0	1	0	0	1	0	12H	–	Partial mode on (PTLON)
0	0	0	0	1	0	0	1	1	13H	–	normal Display mode on (NORON)
0	0	0	1	0	0	0	0	0	20H	–	display inversion off (INVOFF)
0	0	0	1	0	0	0	0	1	21H	–	display inversion on (INVON)
0	0	0	1	0	0	0	1	0	22H	–	all pixel off (DALO)
0	0	0	1	0	0	0	1	1	23H	–	all pixel on (DAL)
0	0	0	1	0	0	1	0	1	25H	–	set contrast (SETCON)
1	X	$VCON_6$	$VCON_5$	$VCON_4$	$VCON_3$	$VCON_2$	$VCON_1$	$VCON_0$	00H	–	set contrast
0	0	0	1	0	1	0	0	0	28H	–	display off (DISPOFF)
0	0	0	1	0	1	0	0	1	29H	–	display on (DISPON)
0	0	0	1	0	1	0	1	0	2AH	–	column address set (CASET)
1	xs[7]	xs[6]	xs[5]	xs[4]	xs[3]	xs[2]	xs[1]	xs[0]	02H	–	X address start; 0 ≤ xs ≤ 83H
1	xe[7]	xe[6]	xe[5]	xe[4]	xe[3]	xe[2]	xe[1]	xe[0]	81H	–	X address end; xs ≤ xe ≤ 83H
0	0	0	1	0	1	0	1	1	2BH	–	page address set (PASET)
1	ys[7]	ys[6]	ys[5]	ys[4]	ys[3]	ys[2]	ys[1]	ys[0]	02H	–	Y address start; 0 ≤ ys ≤ 83H
1	ye[7]	ye[6]	ye[5]	ye[4]	ye[3]	ye[2]	ye[1]	ye[0]	81H	–	Y address end; ys ≤ ye ≤ 83H
0	0	0	1	0	1	1	0	0	2CH	–	memory write (RAMWR)
1	D7	D6	D5	D4	D3	D2	D1	D0	XXH	–	write data
0	0	0	1	0	1	1	0	1	2DH	–	colour set (RGBSET)

Graphics LCD PCF8833: Παραδείγματα εντολών

Set CS =0, DATA =0, CLK =1

Hardware Reset (RS =0 και μετά RS =1)

Set CLK =1, DATA =1

Software Reset Command (0, 0x01)

Sleep Out Command (0, 0x11)

Booster On Command (0,0x03)

Display On Command (0, 0x29)

Normal Display Mode Cmd (0, 0x13)

Set Contrast Command (0, 0x25)

Contrast Value Data (1,0x3F);

Data Order Command (0,0xBA)

Column Address Set Cmd (0,0x2A);

Column Address Start Data (1, 0x00);

Column Address End Data (1, 131);

Παράδειγμα κώδικα για αποστολή εντολής σε PCF8833

```
void sendCMD(char datax) {
  CLK0
  SDA0            //0 δηλώνει Εντολή
  CLK1
 CLK0
  if ((datax &128)!=0) SDA1 else SDA0
  CLK1
  CLK0
  if ((datax &64)!=0) SDA1 else SDA0
  CLK1
  CLK0
    ..............
if ((datax&4)!=0) SDA1 else SDA0
  CLK1
  CLK0
  if ((datax&2)!=0) SDA1 else SDA0
  CLK1
  CLK0
  if ((datax &1)!=0) SDA1 else SDA0
  CLK1
}
```

CLK0 αντιστοιχεί σε
PORTB.F7 = 0
CLK1 αντιστοιχεί σε
PORTB.F7 = 1

SDA0 αντιστοιχεί σε
PORTB.F6 = 0
SDA1 αντιστοιχεί σε
PORTB.F6 = 1

Χρωματισμός Pixel σε PCF8833

BYTE	D/C	7	6	5	4	3	2	1	0
1st write	1	R_3	R_2	R_1	R_0	G_3	G_2	G_1	G_0
2nd write	1	B_3	B_2	B_1	B_0	R_3	R_2	R_1	R_0
3rd write	1	G_3	G_2	G_1	G_0	B_3	B_2	B_1	B_0

Σε Normal Mode, κάθε χρώμα (R,G,B) καθορίζεται με 4 bit.

Χρησιμοποιούνται 3 byte μνήμης για τον καθορισμό των χρωμάτων 2 διαδοχικών pixel

Ζυγό Pixel

Μονό Pixel

Παράδειγμα κώδικα ορισμού χρώματος σε pixel του PCF8833

```
Static char pix;
Static char red, grn, blu;
Static char s1,s2
```

pix: μονό ή ζυγό pixel
Red, grn, blu : bytes για την τιμή του κάθε χρώματος
s1,s2 : bytes για να κρατούν προσωρινά το χρώμα ζυγών pixel

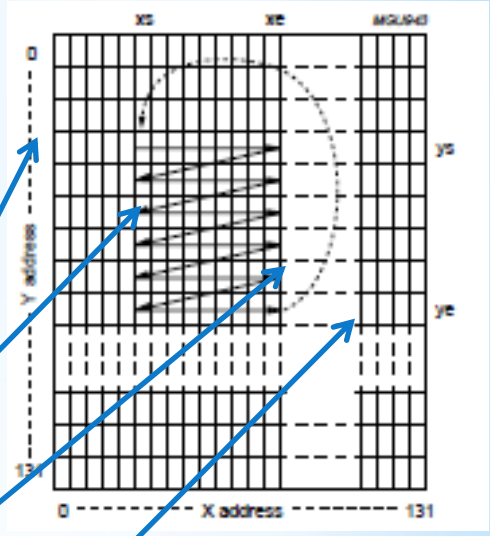
```
void setPixel(char r, char g,char b) {
  red=r_;
  grn=g_;
  blu=b_;
  if (pix==0) {    // even pixel
    s1=(red & 0xF0) | (grn>>4);
    s2=(blu & 0xF0);
    pix=1;
  } else {    //odd pixel
    pix=0;
    sendData(s1);
    sendData(s2|(red>>4));
    sendData((grn & 0xF0) | (blu>>4));
  }

}
```

Αν το pixel είναι ζυγό αποθηκεύουμε προσωρινά το χρώμα σε δύο byte S1,S2 αλλά δεν το στέλνουμε στο display

Αν το pixel είναι μονό στέλνουμε στο display τα τρία Bytes που αντιστοιχούν στα χρώματα των δύο συνεχόμενων pixel.

Παράδειγμα ενεργοποίησης του PCF8833 και ορισμού μέρους της οθόνης .

```
// SET CS BIT TO ZERO.  Επιλογή του Display
 //SEND HARDWARE RESET PULSE
sendCMD(0x11);                  //Sleep Out
sendCMD(0x03);                  //Booster ON
sendCMD(0x29);                  //Display On
sendCMD(0x13);                  //Normal display mode

sendCMD(0x3A);                  //Set color mode
sendData(3);                    //12-Bit per Pixel (default)

sendCMD(0x25);                  //Set Contrast
sendData(63);                   // Contrast value;

sendCMD(0x2A);                   //Column Address Set
 sendData(50);                  //Αρχική στήλη
 sendData(99);                  // Τελική στήλη

sendCMD(0x2B);                  //Page Address Set
 sendData(20);                  //Αρχική σειρά
 sendData(79                    // Τελική σειρά
sendCMD(0x2C);                 //Memory Write  για εγγραφή παραμέτρων στο display
```



Διαδοχικές εγγραφές στην μνήμη ακολουθούν τη σειρά που φαίνεται στο σχήμα.