Médian MC43

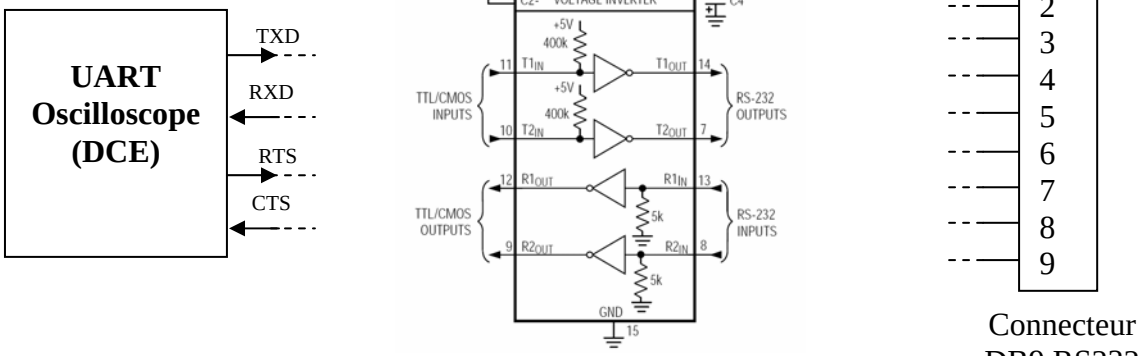
I Transfert de données entre un oscilloscope numérique et un ordinateur PC

On étudie la mise en œuvre de l'acquisition sur un ordinateur PC des données enregistrées par un oscilloscope numérique. L'intérêt est de stocker les relevés de l'oscilloscope et de permettre un traitement et un affichage comparatif des courbes par le PC. Chaque courbe est constituée de 20000 points. Les données sont transférées sous forme ASCII à l'aide du protocole de contrôle de flux matériel RTS – CTS. Le PC est un DTE et l'oscilloscope un DCE. Chaque donnée est envoyée sous forme de chaîne de caractère (suite de codes ASCII terminée par l'octet 0x00). L'oscilloscope envoie dans l'ordre :

- le nombre de points N
- la période d'échantillonnage (base de temps de l'oscilloscope)
- la série des N points dans l'ordre

La liaison série de l'oscilloscope est configurée de la manière suivante : 19200 bauds, données de 7 bits, pas de parité, 1 bit de STOP.

I.1 Compléter le schéma suivant décrivant l'interface matérielle RS232 de l'oscilloscope.



MAX232

Connecteur
DB9 RS232

I.2 Représenter la trame RS232 (signal TXD de l'oscilloscope) en amont et en aval du driver MAX232 lors de l'envoi du caractère '0'.

I.3 L'oscilloscope est relié au port COM1 du PC. Ecrire en langage C le sous-programme ***initCOM1*** permettant d'initialiser le port COM1 du PC pour communiquer avec l'oscilloscope.

I.4 Indiquer quelle suite d'octets correspond à la chaîne de caractères « 20000 ».

I.5 Donner l'organigramme du sous-programme ***recept_char*** réalisant la réception d'un caractère sur la liaison série.

I.6 Ecrire ce sous-programme en langage C.

I.7 Donner l'organigramme du sous-programme ***recept_string*** réalisant la réception d'une chaîne de caractères sur la liaison série.

I.8 Compléter en langage C le sous-programme ***recept_string*** suivant :

```
char *recept_string(void)
  {
  static char s[10];
  …
  return s;
  }
```

I.9 Les données reçues par le PC sont enregistrées dans un fichier. En utilisant les fonctions :

- ***FILE *ouverture_fichier(void)*** : créé et ouvre un fichier sur le disque dur
- ***void fermeture_fichier(FILE *fic)*** : ferme un fichier ouvert
- ***ecriture_fichier(FILE *fic, char *s)*** : écrit une chaîne de caractère dans le fichier ouvert
- ***int atoi(const char *s)*** : convertie une chaîne de caractère en sa valeur entière et retourne cette valeur entière (exemple : si *buf = « 12 », atoi(buf) retourne l'entier 12)

écrire l'organigramme ***recept_oscillo*** réalisant la réception d'un transfert complet par le PC.

I.10 Compléter en langage C le sous-programme ***recept_oscillo*** suivant :

```
void recept_oscillo(void)
  {
  FILE *fic;
  fic = ouverture_fichier();
  …
  …
  fermeture_fichier(fic);
  }
```

I.11 En considérant que chaque chaîne de caractères comprend 6 caractères (caractère de fin de chaîne compris), déterminer le temps de transfert total lorsque N = 20000.

II Instrumentation par bus IEEE-488

On désire mettre en œuvre une chaîne d'instrumentation comprenant un ordinateur PC (contrôleur IEEE-488), un multimètre et un GBF. L'ensemble doit permettre de réaliser la caractérisation de circuits électroniques (réponse fréquentielle), en appliquant des excitations programmables à l'aide du GBF et en relevant les réponses à l'aide du multimètre.

On attribue aux appareils les adresses suivantes :

- PC : adresse = 1
- Multimètre : adresse = 8
- GBF : adresse = 25

II.1 Rappeler quel est le taux de transfert maximal du bus IEEE-488.

II.2 Indiquer pour les 3 appareils quelles valeurs en hexadécimal le contrôleur envoie sur le bus IEEE-488 pour configurer chacun d'entre eux comme « parleur » ou « écouteur ».

II.3 Indiquer comment le contrôleur procède pour désactiver les « parleurs » et les « écouteurs ».

II.4 Rappeler quel est le rôle de la ligne ATN.

III Horloge temps réel I2C : le PCF8563

Un appareil électronique est géré par une carte comprenant un microcontrôleur doté d'un contrôleur I2C interne. Plusieurs composants sont reliés au bus I2C dont l'horloge temps réel PCF8563 dont le rôle est de donner la date et l'heure. Ce composant est alimenté par une pile, ce qui permet de préserver la date et l'heure lors des coupures d'alimentation. Une documentation restreinte du PCF8563 est donnée en annexe. Le microcontrôleur et son contrôleur I2C ne sont pas étudiés ici.

III.1 Indiquer quelles sont les adresses en lecture et en écriture du PCF8563.

III.2 En s'inspirant des exemples fournis dans le datasheet, donner la trame I2C permettant d'initialiser les paramètres suivants de l'horloge temps réel : **Mercredi 03 mai 2006 8h00 (00 secondes)**.

III.3 Donner la trame I2C permettant de lire la date : secondes, minutes, heure, jour du mois, jour de la semaine, mois, siècle (19xx ou 20xx), année.

III.4 En négligeant les temps de traitement du microcontrôleur (accès aux registres et à la mémoire, appels des fonctions), évaluer la durée de lecture de la date lorsque le bus I2C est cadencé à sa vitesse maximale.

PCF8563 : Horloge temps réel

(Extrait du datasheet)

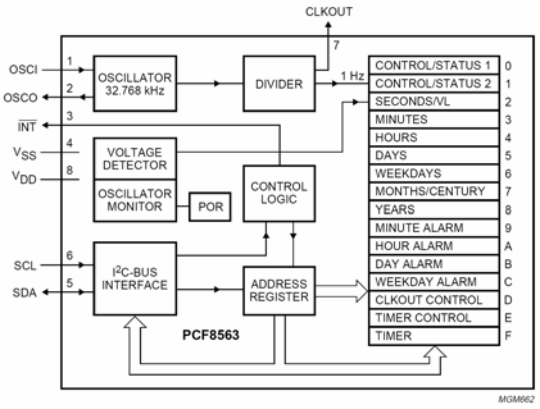
Functional description

The PCF8563 contains sixteen 8-bit registers with an auto-incrementing address register, an on-chip 32.768 kHz oscillator with one integrated capacitor, a frequency divider which provides the source clock for the Real Time Clock/calender (RTC), a programmable clock output, a timer, an alarm, a voltage-low detector and a 400 kHz I²C-bus interface.

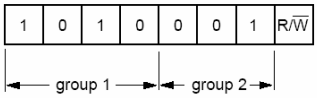
All 16 registers are designed as addressable 8-bit parallel registers although not all bits are implemented. The first two registers (memory address 00H and 01H) are used as control and/or status registers. The memory addresses 02H through 08H are used as counters for the clock function (seconds up to years counters). Address locations 09H through 0CH contain alarm registers which define the conditions for an alarm. Address 0DH controls the CLKOUT output frequency. 0EH and 0FH are the timer control and timer registers, respectively.

The seconds, minutes, hours, days, weekdays, months, years as well as the minute alarm, hour alarm, day alarm and weekday alarm registers are all coded in BCD format.

When one of the RTC registers is read the contents of all counters are frozen. Therefore, faulty reading of the clock/calendar during a carry condition is prevented.



Slave address.



Clock/calendar read/write cycles

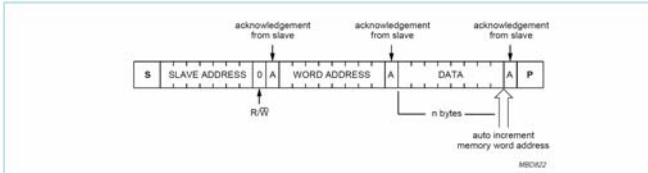


Fig 13. Master transmits to slave receiver (write mode).

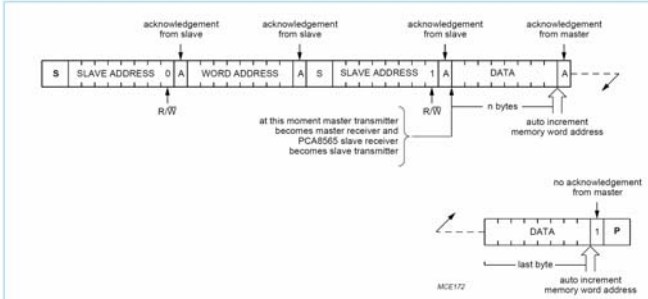


Fig 14. Master reads after setting word address (write word address; read data).

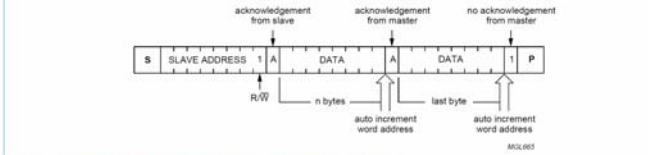


Fig 15. Master reads slave immediately after first byte (read mode).

Register organization

Table 4: Binary formatted registers overview

Bit positions labelled as x are not implemented. Bit positions labelled with 0 should always be written with logic 0; if read they could be either logic 0 or logic 1.

Address	Register name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
00H	control/status 1	TEST1	0	STOP	0	TESTC	0	0	0
01H	control/status 2	0	0	0	TI/TP	AF	TF	AIE	TIE
0DH	CLKOUT control	FE	x	x	x	x	x	FD1	FD0
0EH	timer control	TE	x	x	x	x	x	TD1	TD0
0FH	timer	<timer countdown value>							

Table 5: BCD formatted registers overview

Bit positions labelled as x are not implemented.

Address	Register name	BCD format tens nibble				BCD format units nibble			
		Bit 7 2^3	Bit 6 2^2	Bit 5 2^1	Bit 4 2^0	Bit 3 2^3	Bit 2 2^2	Bit 1 2^1	Bit 0 2^0
02H	seconds	VL	<seconds 00 to 59 coded in BCD>						
03H	minutes	x	<minutes 00 to 59 coded in BCD>						
04H	hours	x	x	<hours 00 to 23 coded in BCD>					
05H	days	x	x	<days 01 to 31 coded in BCD>					
06H	weekdays	x	x	x	x	x	<weekdays 0 to 6>		
07H	months/century	C	x	x	<months 01 to 12 coded in BCD>				
08H	years	<years 00 to 99 coded in BCD>							
09H	minute alarm	AE	<minute alarm 00 to 59 coded in BCD>						
0AH	hour alarm	AE	x	<hour alarm 00 to 23 coded in BCD>					
0BH	day alarm	AE	x	<day alarm 01 to 31 coded in BCD>					
0CH	weekday alarm	AE	x	x	x	x	<weekday alarm 0 to 6>		

Table 9: Seconds/VL (address 02H) bits description

Bit	Symbol	Value	Description
7	VL	0	clock integrity is guaranteed
		1	integrity of the clock information is no longer guaranteed
6 to 0	seconds	00 to 59	this register holds the current seconds coded in BCD format; example: seconds register contains x101 1001 = 59 seconds

Table 14: Weekday assignments

Day	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Sunday	x	x	x	x	x	0	0	0
Monday	x	x	x	x	x	0	0	1
Tuesday	x	x	x	x	x	0	1	0
Wednesday	x	x	x	x	x	0	1	1
Thursday	x	x	x	x	x	1	0	0
Friday	x	x	x	x	x	1	0	1
Saturday	x	x	x	x	x	1	1	0

Table 15: Months/century (address 07H) bits description

Bit	Symbol	Value	Description
7	century[1]		this bit is toggled when the years register overflows from 99 to 00
		0	indicates the century is 20xx
		1	indicates the century is 19xx
4 to 0	month	01 to 12	this register holds the current month coded in BCD format, see Table 16