Exercice sur le Bus I2C

Système CYCLOCITY sur vélo'V

Présentation générale

Le support d'analyse est un système combinant bornes de location et bicyclettes.

Borne d'accueil automatisée

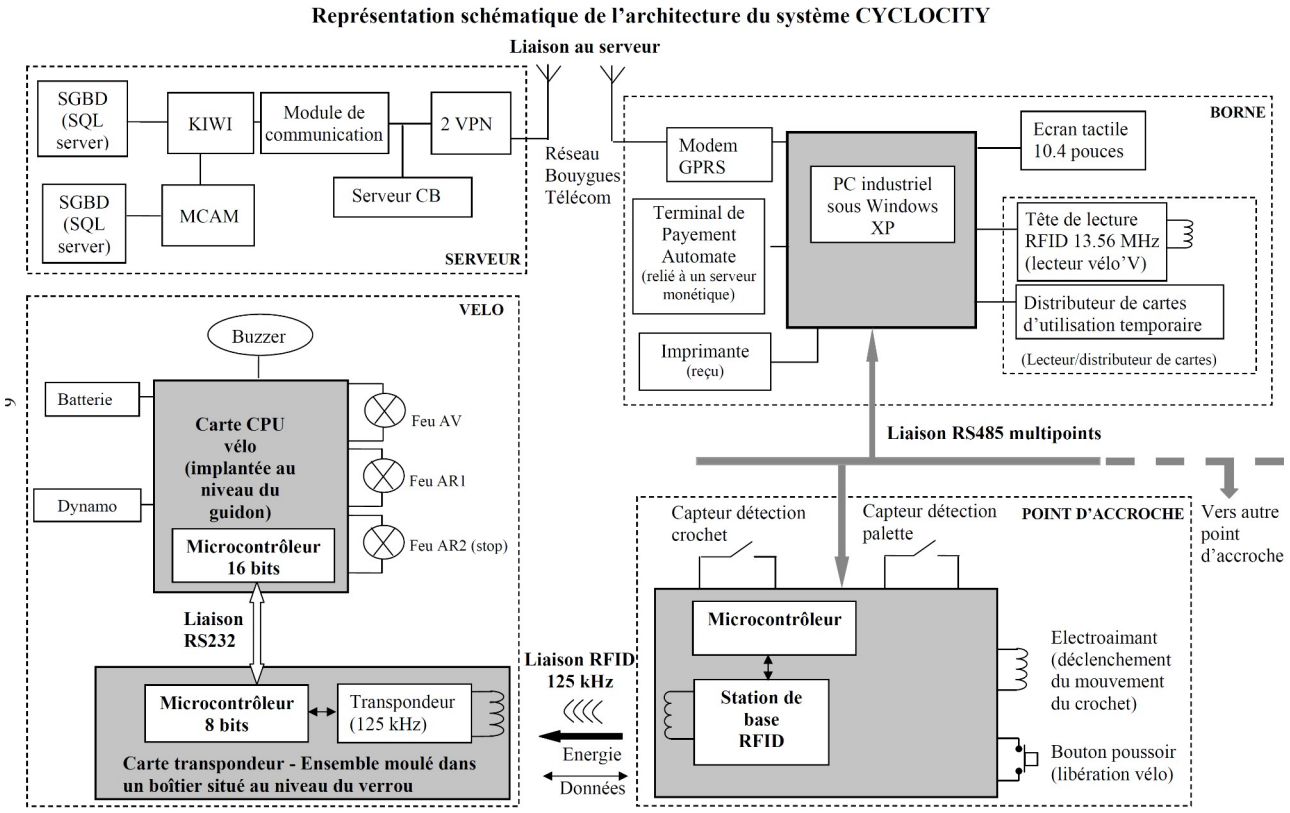
Vélo et point d'attache

La gestion du parc de vélos est rendue possible grâce à l'électronique embarquée à leur bord, (qui réalise des fonctions dédiées à la communication avec les différentes bornes), au stockage d'informations, ainsi qu'à la gestion de l'énergie produite et consommée pendant la course.

Chaque point d'accroche comporte une carte à microprocesseur incluant une station de base RFID capable de dialoguer avec un transpondeur se trouvant sur le vélo par liaison radiofréquence.

Le vélo comporte quant à lui deux cartes électroniques : la première, située au niveau du guidon, comprend un microcontrôleur 16 bits assurant la gestion de l'essentiel des fonctions électroniques réalisées. Nous l'appellerons par la suite **« carte CPU vélo »**. Un boîtier situé à hauteur du verrou du point d'attache renferme la seconde carte sur laquelle se trouvent le transpondeur et le microcontrôleur associé (**« carte transpondeur »**). Ces deux cartes communiquent par liaison série asynchrone RS232.

Lors du retrait du vélo la borne envoie au vélo (via les liaisons RS485 et RFID) les informations suivantes, qui seront stockées dans l'EEPROM du transpondeur :
- identifiant client
- numéro de borne
- numéro de point d'attache
- heure de départ



Questionnaire portant sur le stockage dans la mémoire morte
Le schéma structurel de la « Carte CPU vélo » est joint.

Entourer cette carte sur la représentation schématique de l'architecture du système CYCLOCITY

Un extrait de la documentation de la mémoire est joint.

1. Identifier sur le schéma structurel le circuit mémoire, et indiquer sa référence.
   →

2. Que signifie littéralement EEPROM ?
   →

3. Donner la définition de ce type de mémoire.
   →

4. Combien d'octets (Bytes) contient le circuit ? Préciser le nombre de bits qui sont nécessaires à l'adressage du contenu de la mémoire.
   →
   →

5. Donner la/les adresse(s) de base de l'EEPROM sur le bus I2C telle qu'elle est câblée sur le schéma structurel, en adressage 7 bits, et en mode 8 bits .
   →

6. Préciser le rôle des broches E0, E1 et E2.
   →

7. Quel est le rôle des résistances R76 et R77 ? Quel est normalement l'ordre de grandeur de ces résistances ? *(Voir la notice d'application de Texas Instrument)*
   →
   →

8. Quel est la signification littérale du nom de la broche $\overline{WC}$? Indiquer son rôle et son niveau logique d'activation.
   →
   →
   →

9. Préciser les conditions de début (start) et de fin (stop) des échanges sur le bus I2C. Préciser également les conditions de changement d'état et de validité des données.
   → Start :
   → Stop :
   →

Nous nous intéressons dans ce qui suit au stockage et à la restitution des date et heure de départ du vélo de la borne d'accueil automatisée.

10. Entourer le chronogramme illustrant l'écriture de plusieurs octets consécutifs dans la mémoire. Et expliciter la chronologie.
    →

11. Entourer le chronogramme illustrant la lecture de plusieurs octets consécutifs dans la mémoire. Et expliciter la chronologie.
    →

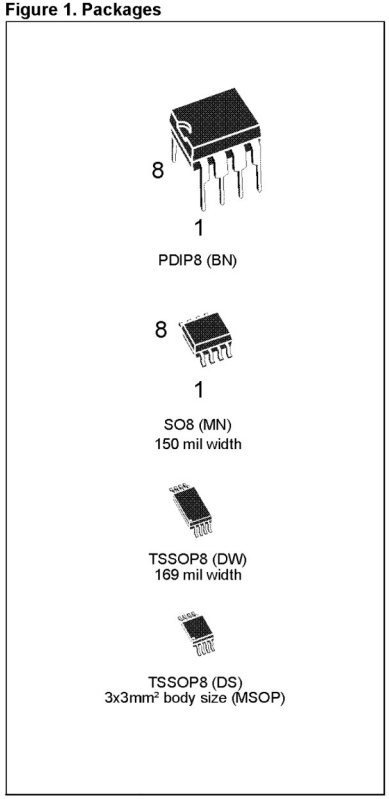
M24C64
M24C32

64Kbit and 32Kbit Serial I²C Bus EEPROM

FEATURES SUMMARY

- Two Wire I^2C Serial Interface Supports 400 kHz Protocol
- Single Supply Voltage:
  - 4.5V to 5.5V for M24Cxx
  - 2.5V to 5.5V for M24Cxx-W
  - 1.8V to 5.5V for M24Cxx-R
- Write Control Input
- BYTE and PAGE WRITE (up to 32 Bytes)
- RANDOM and SEQUENTIAL READ Modes
- Self-Timed Programming Cycle
- Automatic Address Incrementing
- Enhanced ESD/Latch-Up Behavior
- More than 1 Million Erase/Write Cycles
- More than 40 Year Data Retention

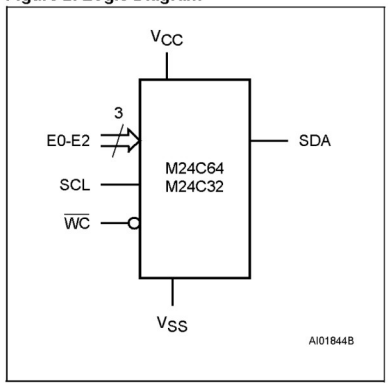
Figure 1. Packages



SUMMARY DESCRIPTION

These I^2C-compatible electrically erasable programmable memory (EEPROM) devices are organized as 8192 x 8 bits (M24C64) and 4096 x 8 bits (M24C32).

Figure 2. Logic Diagram



These devices are compatible with the I^2C memory protocol. This is a two wire serial interface that uses a bi-directional data bus and serial clock. The devices carry a built-in 4-bit Device Type Identifier code (1010) in accordance with the I^2C bus definition.

The device behaves as a slave in the I^2C protocol, with all memory operations synchronized by the serial clock. Read and Write operations are initiated by a Start condition, generated by the bus master. The Start condition is followed by a Device Select Code and $\overline{RW}$ bit (as described in Table 2), terminated by an acknowledge bit.

When writing data to the memory, the device inserts an acknowledge bit during the 9^{th} bit time, following the bus master's 8-bit transmission. When data is read by the bus master, the bus master acknowledges the receipt of the data byte in the same way. Data transfers are terminated by a Stop condition after an Ack for Write, and after a NoAck for Read.

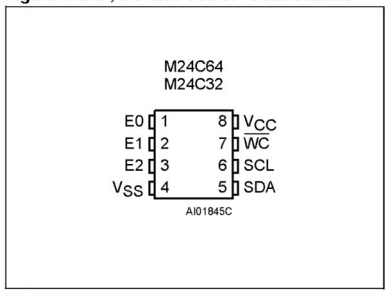
Table 1. Signal Names

E0, E1, E2	Chip Enable
SDA	Serial Data
SCL	Serial Clock
$\overline{WC}$	Write Control
V_{CC}	Supply Voltage
V_{SS}	Ground

Power On Reset: V_{CC} Lock-Out Write Protect

In order to prevent data corruption and inadvertent Write operations during Power-up, a Power On Reset (POR) circuit is included. The internal reset is held active until V_{CC} has reached the POR threshold value, and all operations are disabled – the device will not respond to any command. In the same way, when V_{CC} drops from the operating voltage, below the POR threshold value, all operations are disabled and the device will not respond to any command. A stable and valid V_{CC} must be applied before applying any logic signal.

Figure 3. DIP, SO and TSSOP Connections



Note: 1. See page 18 (onwards) for package dimensions, and how to identify pin-1.

SIGNAL DESCRIPTION

Serial Clock (SCL)

This input signal is used to strobe all data in and out of the device. In applications where this signal is used by slave devices to synchronize the bus to a slower clock, the bus master must have an open drain output, and a pull-up resistor must be connected from Serial Clock (SCL) to V_{CC}. (Figure 4 indicates how the value of the pull-up resistor can be calculated). In most applications, though, this method of synchronization is not employed, and so the pull-up resistor is not necessary, provided that the bus master has a push-pull (rather than open drain) output.

Serial Data (SDA)

This bi-directional signal is used to transfer data in or out of the device. It is an open drain output that may be wire-OR'ed with other open drain or open collector signals on the bus. A pull up resistor must be connected from Serial Data (SDA) to V_{CC}. (Figure 4 indicates how the value of the pull-up resistor can be calculated).

Chip Enable (E0, E1, E2)

These input signals are used to set the value that is to be looked for on the three least significant bits (b3, b2, b1) of the 7-bit Device Select Code. These inputs must be tied to V_{CC} or V_{SS}, to establish the Device Select Code.

Write Control ($\overline{WC}$)

This input signal is useful for protecting the entire contents of the memory from inadvertent write operations. Write operations are disabled to the entire memory array when Write Control ($\overline{WC}$) is driven High. When unconnected, the signal is internally read as V_{IL}, and Write operations are allowed.

When Write Control ($\overline{WC}$) is driven High, Device Select and Address bytes are acknowledged, Data bytes are not acknowledged.

Figure 4. Maximum R_L Value versus Bus Capacitance (C_{BUS}) for an I^2C Bus

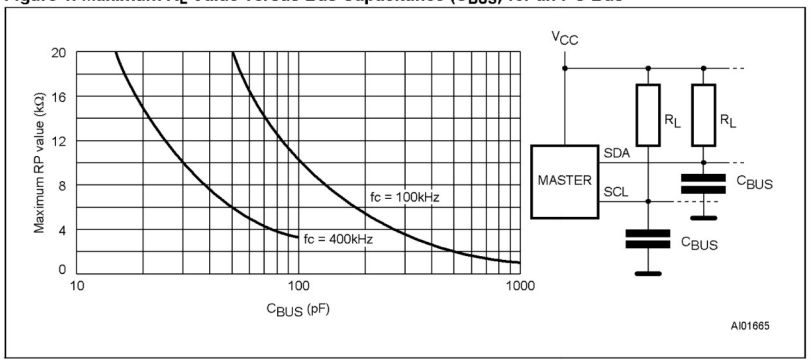


Figure 5. I^2C Bus Protocol

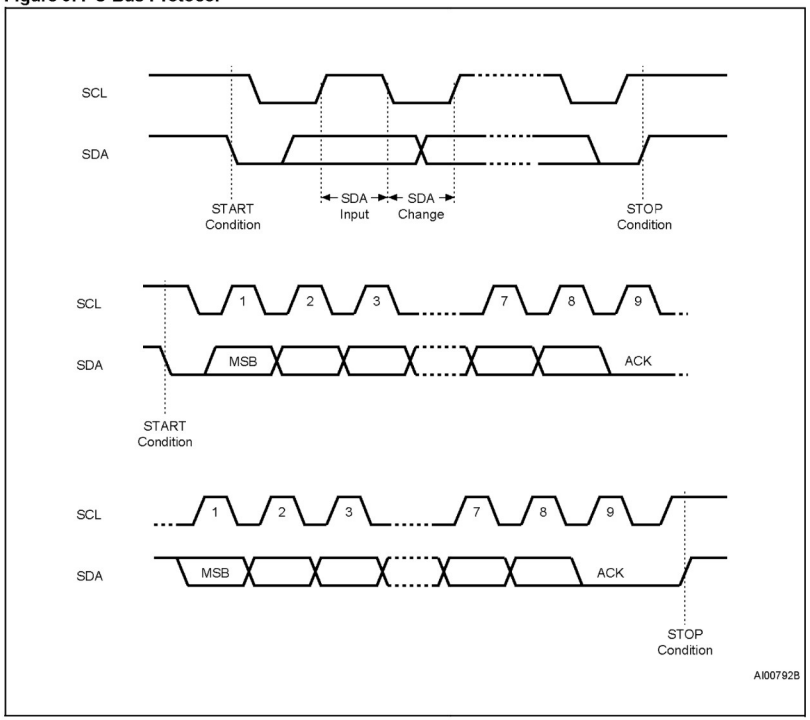


Table 2. Device Select Code

	Device Type Identifier[1]				Chip Enable Address[2]			$\overline{RW}$
	b7	b6	b5	b4	b3	b2	b1	b0
Device Select Code	1	0	1	0	E2	E1	E0	$\overline{RW}$

Note: 1. The most significant bit, b7, is sent first.
2. E0, E1 and E2 are compared against the respective external pins on the memory device.

Table 3. Most Significant Byte

b15	b14	b13	b12	b11	b10	b9	b8

Table 4. Least Significant Byte

b7	b6	b5	b4	b3	b2	b1	b0

DEVICE OPERATION

The device supports the I^2C protocol. This is summarized in Figure 5. Any device that sends data on to the bus is defined to be a transmitter, and any device that reads the data to be a receiver. The device that controls the data transfer is known as the bus master, and the other as the slave device. A data transfer can only be initiated by the bus master, which will also provide the serial clock for synchronization. The M24Cxx device is always a slave in all communication.

Start Condition

Start is identified by a falling edge of Serial Data (SDA) while Serial Clock (SCL) is stable in the High state. A Start condition must precede any data transfer command. The device continuously monitors (except during a Write cycle) Serial Data (SDA) and Serial Clock (SCL) for a Start condition, and will not respond unless one is given.

Stop Condition

Stop is identified by a rising edge of Serial Data (SDA) while Serial Clock (SCL) is stable and driven High. A Stop condition terminates communication between the device and the bus master. A Read command that is followed by NoAck can be followed by a Stop condition to force the device into the Stand-by mode. A Stop condition at the end of a Write command triggers the internal EEPROM Write cycle.

Acknowledge Bit (ACK)

The acknowledge bit is used to indicate a successful byte transfer. The bus transmitter, whether it be bus master or slave device, releases Serial Data (SDA) after sending eight bits of data. During the 9th clock pulse period, the receiver pulls Serial Data (SDA) Low to acknowledge the receipt of the eight data bits.

Data Input

During data input, the device samples Serial Data (SDA) on the rising edge of Serial Clock (SCL). For correct device operation, Serial Data (SDA) must be stable during the rising edge of Serial Clock (SCL), and the Serial Data (SDA) signal must change *only* when Serial Clock (SCL) is driven Low.

Memory Addressing

To start communication between the bus master and the slave device, the bus master must initiate a Start condition. Following this, the bus master sends the Device Select Code, shown in Table 2 (on Serial Data (SDA), most significant bit first).

The Device Select Code consists of a 4-bit Device Type Identifier, and a 3-bit Chip Enable "Address" (E2, E1, E0). To address the memory array, the 4-bit Device Type Identifier is 1010b.

Up to eight memory devices can be connected on a single I^2C bus. Each one is given a unique 3-bit code on the Chip Enable (E0, E1, E2) inputs. When the Device Select Code is received on Serial Data (SDA), the device only responds if the Chip Enable Address is the same as the value on the Chip Enable (E0, E1, E2) inputs.

The 8th bit is the Read/$\overline{\text{Write}}$ bit ($\overline{RW}$). This bit is set to 1 for Read and 0 for Write operations.

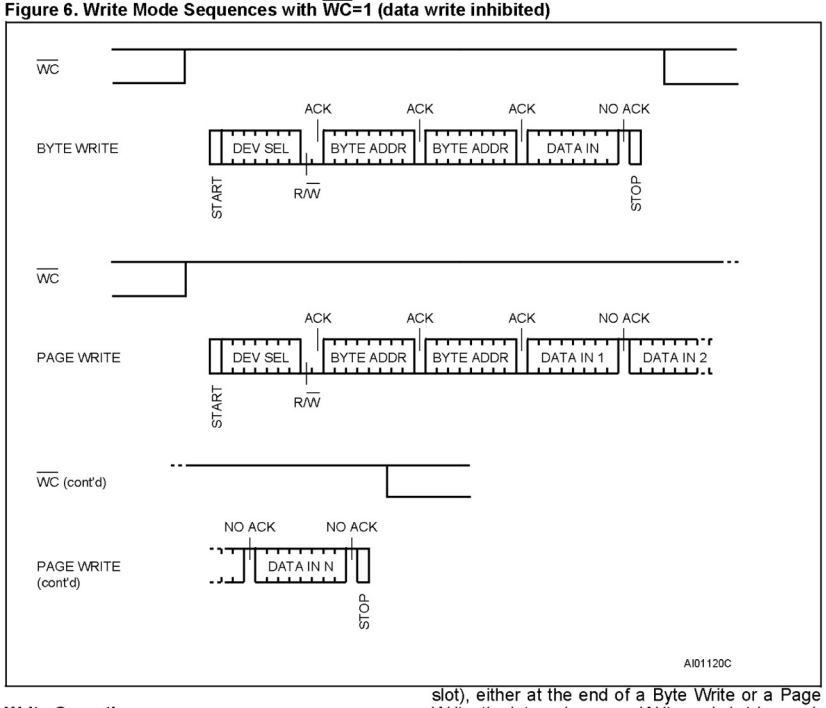
If a match occurs on the Device Select code, the corresponding device gives an acknowledgment on Serial Data (SDA) during the 9th bit time. If the device does not match the Device Select code, it deselects itself from the bus, and goes into Stand-by mode.

Table 5. Operating Modes

Mode	$\overline{RW}$ bit	$\overline{WC}$ [1]	Bytes	Initial Sequence
Current Address Read	1	X	1	START, Device Select, $\overline{RW}$ = 1
Random Address Read	0	X	1	START, Device Select, $\overline{RW}$ = 0, Address
	1	X		reSTART, Device Select, $\overline{RW}$ = 1
Sequential Read	1	X	≥ 1	Similar to Current or Random Address Read
Byte Write	0	V_{IL}	1	START, Device Select, $\overline{RW}$ = 0
Page Write	0	V_{IL}	≤ 32	START, Device Select, $\overline{RW}$ = 0

Note: 1. X = V_{IH} or V_{IL}.

Figure 6. Write Mode Sequences with $\overline{WC}$=1 (data write inhibited)



Write Operations

Following a Start condition the bus master sends a Device Select Code with the $\overline{RW}$ bit reset to 0. The device acknowledges this, as shown in Figure 7, and waits for two address bytes. The device responds to each address byte with an acknowledge bit, and then waits for the data byte.

Writing to the memory may be inhibited if Write Control ($\overline{WC}$) is driven High. Any Write instruction with Write Control ($\overline{WC}$) driven High (during a period of time from the Start condition until the end of the two address bytes) will not modify the memory contents, and the accompanying data bytes are *not* acknowledged, as shown in Figure 6.

Each data byte in the memory has a 16-bit (two byte wide) address. The Most Significant Byte (Table 3) is sent first, followed by the Least Significant Byte (Table 4). Bits b15 to b0 form the address of the byte in memory.

When the bus master generates a Stop condition immediately after the Ack bit (in the "10th bit" time slot), either at the end of a Byte Write or a Page Write, the internal memory Write cycle is triggered. A Stop condition at any other time slot does not trigger the internal Write cycle.

During the internal Write cycle, Serial Data (SDA) is disabled internally, and the device does not respond to any requests.

Byte Write

After the Device Select code and the address bytes, the bus master sends one data byte. If the addressed location is Write-protected, by Write Control ($\overline{WC}$) being driven High, the device replies with NoAck, and the location is not modified. If, instead, the addressed location is not Write-protected, the device replies with Ack. The bus master terminates the transfer by generating a Stop condition, as shown in Figure 7.

Page Write

The Page Write mode allows up to 32 bytes to be written in a single Write cycle, provided that they are all located in the same 'row' in the memory: that is, the most significant memory address bits

(b12-b5 for M24C64, and b12-b5 for M24C32) are the same. If more bytes are sent than will fit up to the end of the row, a condition known as 'roll-over' occurs. This should be avoided, as data starts to become overwritten in an implementation dependent way.

The bus master sends from 1 to 32 bytes of data, each of which is acknowledged by the device if Write Control ($\overline{WC}$) is Low. If Write Control ($\overline{WC}$) is High, the contents of the addressed memory location are not modified, and each data byte is followed by a NoAck. After each byte is transferred, the internal byte address counter (the 5 least significant address bits only) is incremented. The transfer is terminated by the bus master generating a Stop condition.

Figure 7. Write Mode Sequences with $\overline{WC}$=0 (data write enabled)

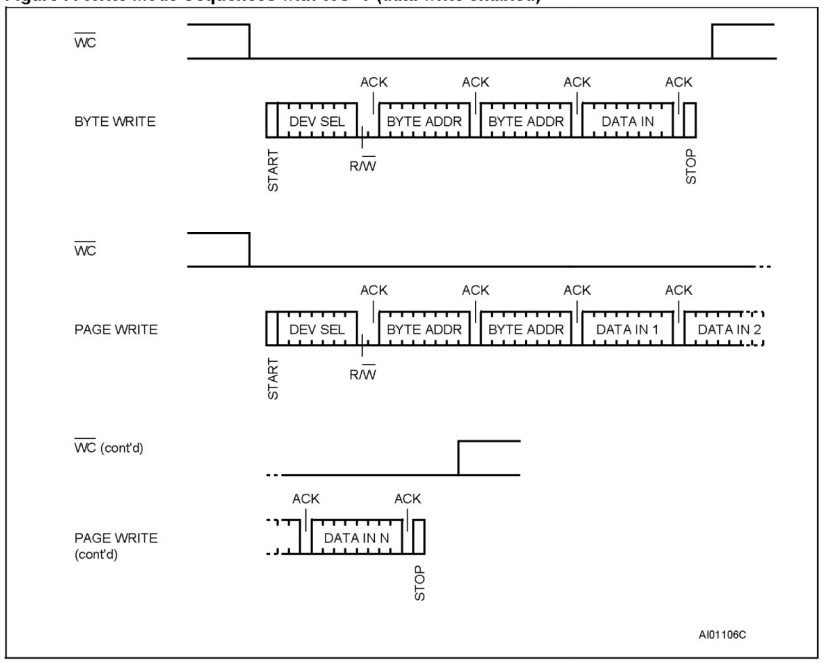
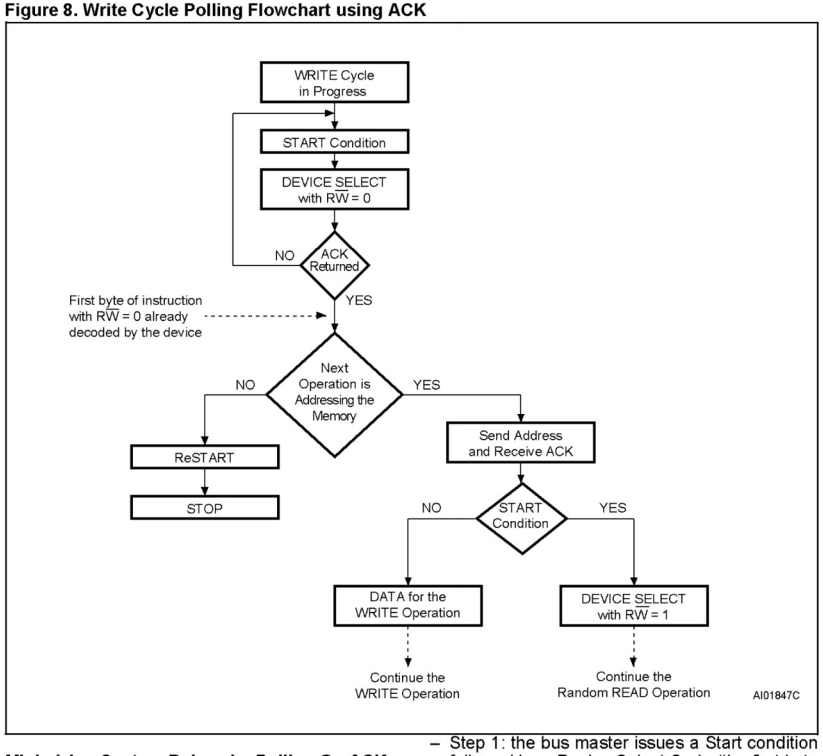


Figure 8. Write Cycle Polling Flowchart using ACK



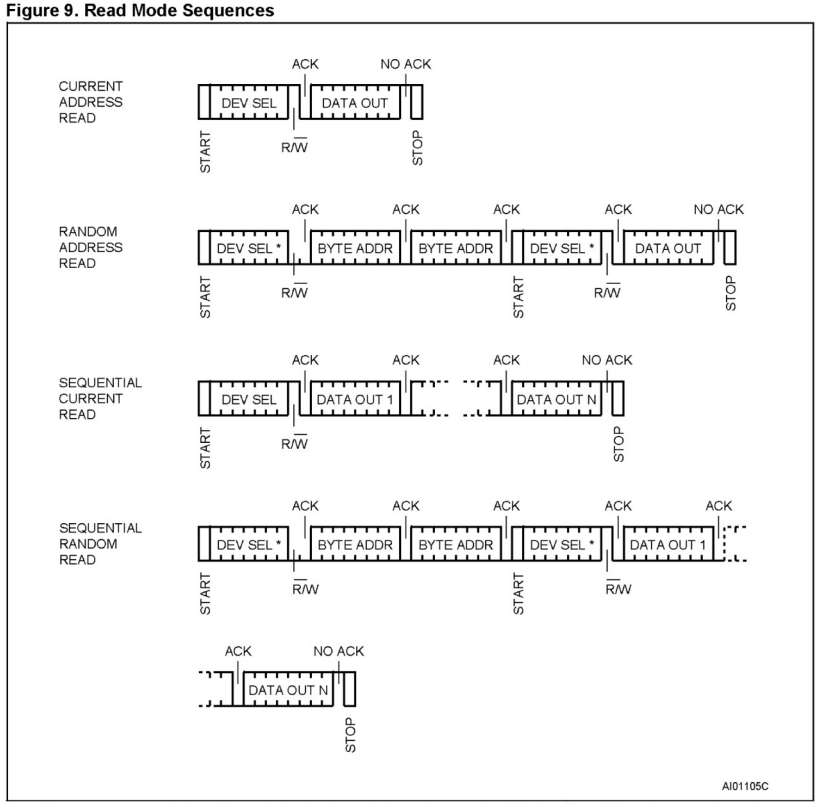
Minimizing System Delays by Polling On ACK

During the internal Write cycle, the device disconnects itself from the bus, and writes a copy of the data from its internal latches to the memory cells. The maximum Write time (t_W) is shown in Tables 17 and 18, but the typical time is shorter. To make use of this, a polling sequence can be used by the bus master.

The sequence, as shown in Figure 8, is:

– Initial condition: a Write cycle is in progress.
– Step 1: the bus master issues a Start condition followed by a Device Select Code (the first byte of the new instruction).
– Step 2: if the device is busy with the internal Write cycle, no Ack will be returned and the bus master goes back to Step 1. If the device has terminated the internal Write cycle, it responds with an Ack, indicating that the device is ready to receive the second part of the instruction (the first byte of this instruction having been sent during Step 1).

Figure 9. Read Mode Sequences



Note: 1. The seven most significant bits of the Device Select Code of a Random Read (in the 1st and 4th bytes) must be identical.

Read Operations

Read operations are performed independently of the state of the Write Control ($\overline{WC}$) signal.

Random Address Read

A dummy Write is performed to load the address into the address counter (as shown in Figure 9) but *without* sending a Stop condition. Then, the bus master sends another Start condition, and repeats the Device Select Code, with the $\overline{RW}$ bit set to 1. The device acknowledges this, and outputs the contents of the addressed byte. The bus master must *not* acknowledge the byte, and terminates the transfer with a Stop condition.

Current Address Read

The device has an internal address counter which is incremented each time a byte is read. For the Current Address Read operation, following a Start condition, the bus master only sends a Device Select Code with the $\overline{RW}$ bit set to 1. The device acknowledges this, and outputs the byte addressed by the internal address counter. The counter is then incremented. The bus master terminates the transfer with a Stop condition, as shown in Figure 9, *without* acknowledging the byte.

Sequential Read

This operation can be used after a Current Address Read or a Random Address Read. The bus master *does* acknowledge the data byte output, and sends additional clock pulses so that the device continues to output the next byte in sequence. To terminate the stream of bytes, the bus master must *not* acknowledge the last byte, and *must* generate a Stop condition, as shown in Figure 9.

The output data comes from consecutive addresses, with the internal address counter automatically incremented after each byte output. After the last memory address, the address counter 'rolls-over', and the device continues to output data from memory address 00h.

Acknowledge in Read Mode

For all Read commands, the device waits, after each byte read, for an acknowledgment during the 9th bit time. If the bus master does not drive Serial Data (SDA) Low during this time, the device terminates the data transfer and switches to its Stand-by mode.