Electronic
components
and materials

PHILIPS

D^{2}B specification

Archief
Ir. A.P.M.Moelands

CONTENTS

1.0 INTRODUCTION

2.0 D^2B CONCEPT

3.0 GENERAL CHARACTERISTICS

4.0 ARBITRATION

5.0 MESSAGE PROTOCOL

- 5.1 Start bit field
- 5.2 Mode field
- 5.3 Master field
- 5.4 Slave field
- 5.5 Control field
- 5.6 Data field
  - 5.6.1 Write action on the bus
  - 5.6.2 Read action on the bus

6.0 BIT PROTOCOL

- 6.1 Bit formats
  - 6.1.1 Start bit
  - 6.1.2 Arbitrating bit
  - 6.1.3 Master to slave bit
  - 6.1.4 Slave to master bit

7.0 BIT USAGE

- 7.1 Control bits
- 7.2 End-of-data bit
- 7.3 Parity bit
- 7.4 Acknowledge bit

8.0 ELECTRICAL SPECIFICATIONS

9.0 TIMING

9.1 Bit timing

9.1.1 Start bit timing
9.1.2 Mode bit timing
9.1.3 Master address bit timing
9.1.4 Master to slave bit timing
9.1.5 Slave to master bit timing

9.2 Message times and transmission speeds

SUPPLEMENT 1

Data interpretation:

1.0 SLAVE STATUS
2.0 LOCK ADDRESS
3.0 MEMORY ADDRESS
4.0 DATA
5.0 COMMANDS

5.1 Command codes applicable to the address space 256 - 511

5.1.1 Command codes applicable to any type of equipment
5.1.2 Command codes applicable to specified types of equipment

5.1.2.1 Recorders and players
5.1.2.2 Tuners (Video & Audio)
5.1.2.3 Source selector

5.2 Reserved codes for future standardization
5.3 Command procedures

SUPPLEMENT 2

Address allocation:

1.0 ADDRESS SPACE DEFINITION
2.0 ALLOCATION OF ADDRESS CODES

Archief
Ir. A.P.M.Moelands

1.0 INTRODUCTION

The use of microprocessors and microcomputers within various types of stand-alone domestic, business and industrial equipment has led to a demand for this 'isolated intelligence' to be linked.

- In the home: most modern TVs, VCRs and Hi-Fis include microcomputer-control, yet cannot control each other. For example, when turning on a VCR to play-back a recording, the TV should also be automatically turned on and retuned to the VCR setting.

- In the office: word processors, small business computers and telex machines are usually stand-alone systems, although each possesses information that the other needs. And, if stand-alone systems become overloaded, expansion is difficult, if not impossible.

- In industrial, or automotive, applications: many different control and monitoring devices are generally implemented on one system (the motor car is a prime example). All these controls need to communicate, since one change in system status usually causes others. However, it is very important that this communication should not affect the performance of the controlling device.

Various solutions have been suggested to combat the problem of "isolated intelligence". In large businesses requiring units within an office block, or even a town, to communicate, the Local Area Network (LAN) is gaining favour.

In Local Area Networks, any unit can take control of the serial data channel and transmit data to any other unit. LANs can expand modularly, adapt to changing traffic patterns, and usually continue to operate when one of the units fails. Typical LANs contain hundreds of work stations and transmit data at rates somewhere between 1 and 10 Mbit/s, over distances ranging from 400 m to over 1 km.

However, this performance comes at a high price, in the form of complex protocols, controllers and system software; and for larger systems, the need for professional network managers. Obviously, the LAN is over-specified for the examples we have already mentioned. These applications only require enough performance to handle the throughput of a modest number of microcomputer-controlled units.

To match network capacity to actual application requirements, Philips has introduced two serial buses, the D^2B (Digital Data Bus) and I^2C (Inter-IC) bus (Fig. 1.1).

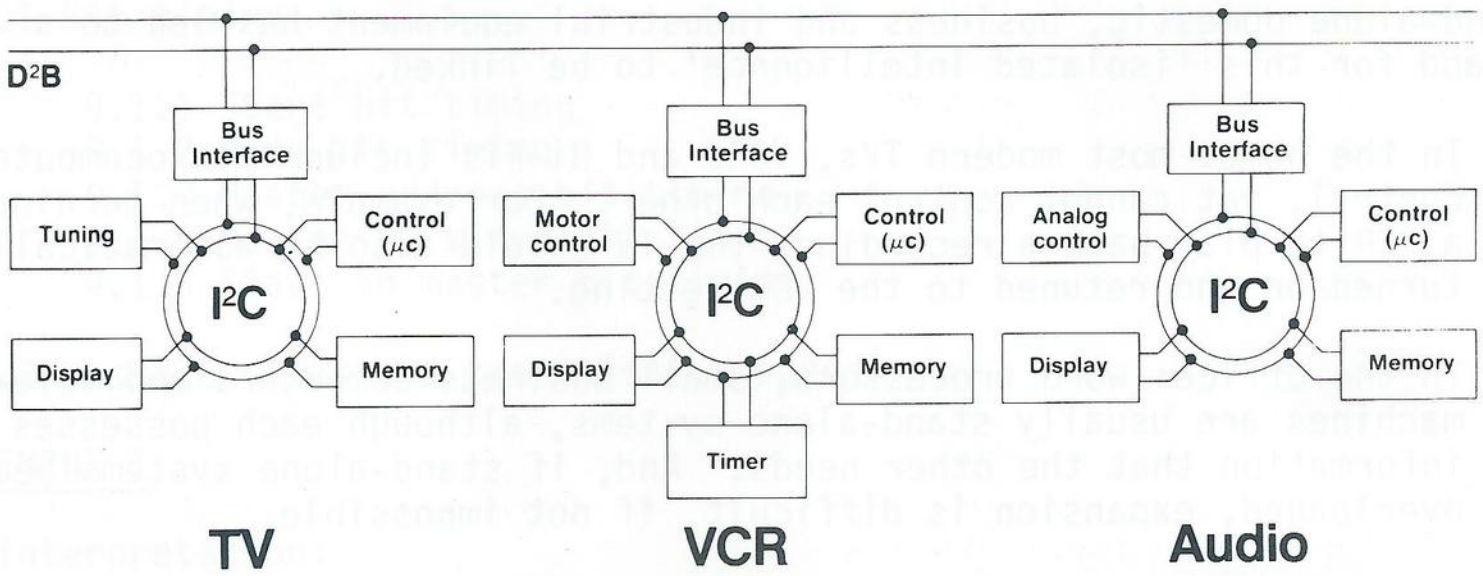


Fig. 1.1 The D²B concept applied to an integrated Video/Audio system, each part of which employs the I²C bus.

The I²C bus connects various devices within a unit, or on a circuit board. It provides an easy means of upgrading equipment by enabling devices to be added or removed as necessary. Wiring is also reduced from the 8 wires for typical parallel data transmissions to a 2-wire serial link. This feature also saves space and simplifies circuit board layout.

The Digital Data Bus (D²B) connects equipment rather than integrated circuits. In order to minimize interference caused by energy radiation, the D²B is implemented as a differentially-driven cable pair. Changes of state are therefore represented as the difference between two (non-TTL level) voltages.

The D²B complements both the LAN and I²C bus concepts. It can perform both as an integrated circuit communication path and as a small LAN-type network. Therefore, the D²B is the link, between I²C bus and LAN performance, that allows a hierarchy of network data communications:

1) I²C bus - communication within equipment

   (100 kbit/s max., distance ~ a few metres max.).

2) D²B - communication between equipment in close proximity

   (100kbits/s max., resulting in an effective character rate of 7.8 kbytes/s., distance ~ 150 m max.).

3) LAN - communication between widely distributed equipment

   (~ 10 Mbits/s max., distance ~ 1 km max.).

2.0 D²B CONCEPT

The following requirements were put forward for the D²B:

- The bus should allow any unit capable of taking control of the bus to do so. It should also permit control of one unit by another.

- The bus should allow a self-configuring network to be built up.

The bus was also required to handle the following operational constraints:

- The removal of a unit should not affect the functioning of the remaining units (nor should power switching on/off in a unit).

- The bus should fit into the audio/video Euroconnector proposal (maximum of 2 signal wires available).

- Interference from the bus should not degrade the proper functioning of equipment in its vicinity.

The D²B is designed to perform in a number of different environments (industrial, domestic, automotive, data processing) and can transfer data at rates appropriate to each application area. There are several transfer modes. At present, three modes have been defined, but more can be introduced if they are needed.

Mode 0: Nominal transfer rate up to 209 char/s. The controller in this mode can operate with an internal clock tolerance of ± 25%, permitting the use of an RC network. Mode 0 is designed to also allow a microprocessor (with a crystal-controlled clock) which is not fully occupied with other tasks, to act as a bus controller.

Mode 1: Nominal transfer rate up to 2457char/s. This mode allows an inexpensive hardware design for the bus controller. The controller can work with a high internal clock tolerance of ± 25%, permitting the use of an RC network rather than a crystal.

Mode 2: Nominal transfer rate up to 7760 char/s. The controller for this mode allows the use of a clock with a tolerance of ± 0,5%, permitting the use of a ceramic resonator.

These modes can be mixed within the same system. For example, a printer may be able to operate in mode 1 and a disc drive in mode 2, with a processing unit switching from one mode to the other to communicate with each of these peripherals. Thus any "intelligent" unit connected to the bus can communicate with any other unit.

On the D²B, units perform as masters and slaves during data transfers. A master unit is any device, such as a microcomputer-controlled VCR, which is capable of initiating and controlling a data transfer.

Depending on the direction of transfer, units also perform as transmitters or receivers. To use the previous example, both the disc drive and the printer are slave devices. The disc drive, however, can both receive and transmit data, whilst the printer can usually only receive.

Archief
Ir. A.P.M.Moelands

The D²B is a multi-master bus, which means that any device capable of controlling the bus can do so, without disturbing the operation of the bus.

When a transfer is initiated, the unit(s) wishing to use the bus lays a claim by generating a unique condition - the start bit. If more than one unit tries to take control of the bus during this bit, their claims are settled in subsequent bits by an arbitration protocol.

The protocol stems from the wired-AND connection of each unit to the D²B. Consequently, there is no central master, or exchange, required.

To try and establish the right to use the bus, each competing unit transmits the binary code associated with the mode in which it wants to work - lower modes have higher priority. If the mode is the same, arbitration continues to the next stage. In this stage, each unit transmits its own address.

Since each unit connected to the D²B has a unique address, there will be a difference between the bits transmitted by competing masters, and the first to generate a "1" when the other generates a "0" will lose the arbitration (for more information on arbitration, see section 4.0).

The unit which wins the arbitration is now the master and controls the bus. So that one master cannot monopolize the bus, it is only allowed to control the bus for one interval, or time slot.

There are 6 bit-fields (Fig. 2.1). Three of these fields; the start, mode and master address fields have already been described; they deal with the initialization and arbitration involved in a transfer. The remaining fields are the slave address, control and data fields.

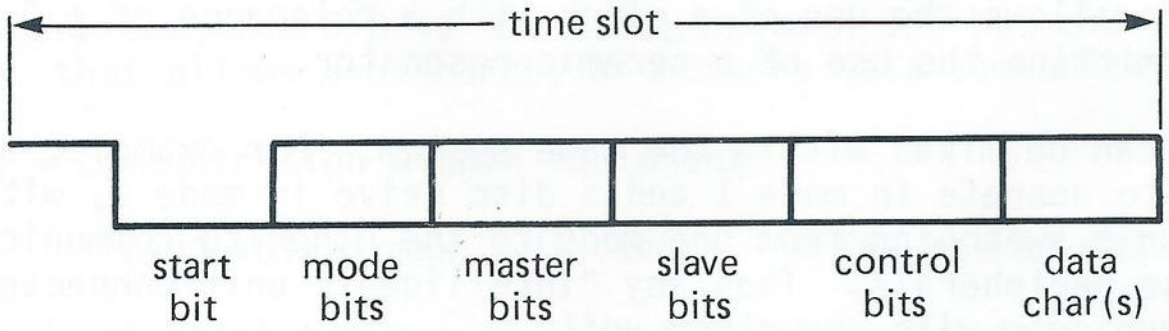


Fig. 2.1 Message format of the D²B, showing the 6 bit-fields.

When a master has gained control of the bus, it sends the address of the unit it wants as a slave. Control bits are then sent to indicate the direction and nature of the transfer. Finally, the data bytes are either sent or received. A receiver-generated acknowledge is also necessary to indicate the correct reception of the slave address, control bits and each data byte during the transmission.

In modes 1 and 2, the maximum number of bytes transferred depends on whether the transfer is master to slave, or slave to master. In a master to slave transfer, timing is determined by the master only - this means that the master generates synchronization and data for each bit transferred. In a slave to master transfer, the master still generates the synchronization but the slave puts the data value on to the bus. Thus, slave to master timing is determined by the combination of both units and, as a consequence, is longer (more information on bit formats will be given in section 5.0).

The effect of finite propagation times and rise and fall times have been taken into account in calculating the number of clock pulses required for each bit field. In addition, the influence of the necessary driving and decoding logic imposes constraints on the timing. These constraints lead to the data transmission capabilities of modes 1 and 2 depending on the direction of transfer.

The number of data bytes that can be sent in one time slot are:

- in mode 0: 2 data bytes in either direction
- in mode 1: 32 data bytes from master to slave
  16 data bytes from slave to master
- in mode 2: 128 data bytes from master to slave
  64 data bytes from slave to master

A unit may use the bus for less than one time slot, but never for more. If a master still needs to transmit or receive information after its time slot expires, it must again try for control of the bus.

However, there are cases when a master cannot afford to lose control of a slave. For example, a master may have set a register within a slave before sending data; if the value of this register is changed in a subsequent time slot, the master cannot resume the transfer when it again controls the bus.

To avoid this situation, the master can request the slave to be locked to it. This is achieved by sending the appropriate control bits in the control field. Other masters are then restricted from accessing that slave until it is unlocked.

Archief
Ir. A.P.M.Moelands

3.0 GENERAL CHARACTERISTICS

The D²B requires only one data channel. In this specification it is defined in terms of a differentially-driven cable pair - but the D²B can use many types of data channel as long as it can be in a wired-AND configuration.

The maximum length of the D²B as a differentially-driven cable pair, which may be twisted to give higher noise immunity, is 150 m and up to 50 units can be connected to it. The maximum number of units, due to the capacity of the addressing allocation, that can be connected to the bus is 4096. The D²B can be used in a bus configuration and/or in a star network.

4.0 ARBITRATION

Every unit wishing to control the D²B has to go through an arbitration procedure. This procedure involves comparing the mode and master address bits of arbitrating-masters (Fig. 4.1), using the wired-AND connection of every unit to the D²B.

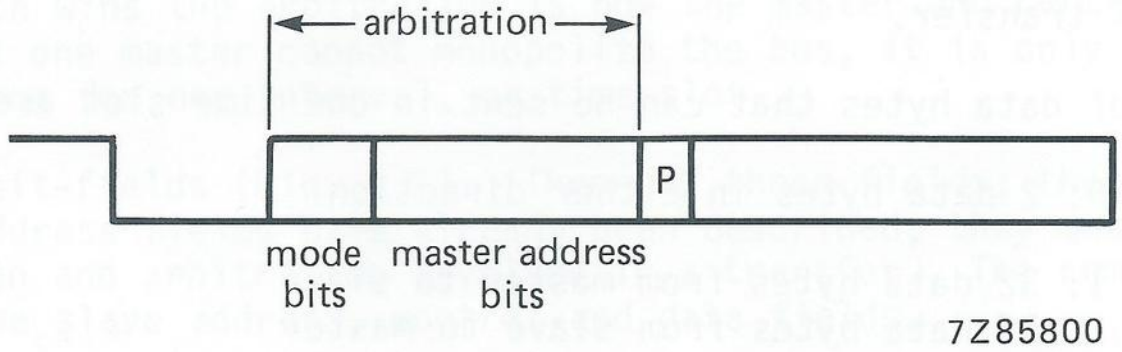


Fig. 4.1 Arbitration takes place during the mode and master fields.

- After each mode bit is placed on the bus, and no other master in a lower mode is found (i.e. the mode bit was a '1'), the masters raise their transmission speed and arbitrate again. Therefore, at the end of the mode bits, the masters that remain are competing in the same mode.

- For each arbitration bit, each arbitrating-master has to check that the actual level on the bus corresponds to the level it placed on the bus. If it doesn't, the arbitrating-master has lost the arbitration and has to follow the rest of the message as a possible slave. At the end of the master address bits, there is only one master.

Due to the wired-AND property of connections to the D²B, a '0' put on the bus will override a '1', this results in the following priority during arbitration:

- the lowest mode has the highest priority
- in the case of an equal mode, the lowest master address has the highest priority.

5.0 MESSAGE PROTOCOL

This section explains how individual bits fit into fields, and what these fields mean in the generation of a message on the D^2B (Fig. 5.1).

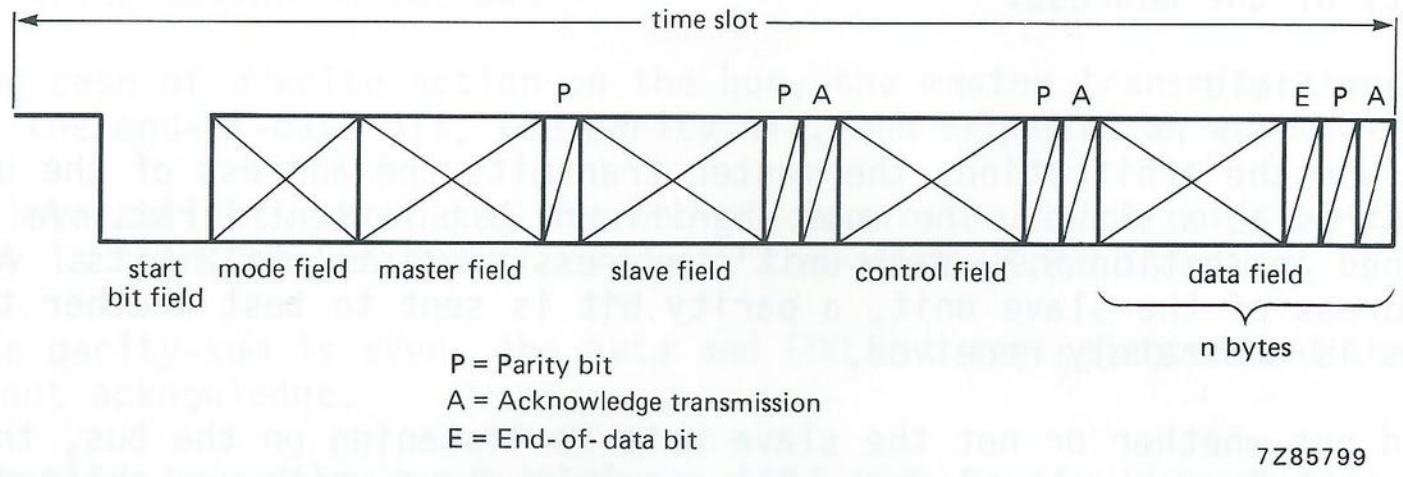


Fig. 5.1 A complete message on the D^2B.

5.1 Start bit field

A unit that wants to master the bus tests whether it is in use and attempts to capture it. If the unit does not want to become a master it monitors the bus.

To initiate a message transfer, the unit puts a LOW on the bus for a given, unique, length of time. The other units will recognize this as the start bit. At the end of this period masters competing for the bus enter the mode field. Slaves continue monitoring the bus to see if they are being addressed. If the bit is too short for a start bit then everyone returns to the monitoring state.

5.2 Mode field

On the D^2B, three different speed modes can be selected by the mode bits. At the end of the start bit competing masters (which may be in different speed modes) enter the mode field by putting a series of bits on the bus.

Mode 0 = 0
Mode 1 = 10
Mode 2 = 110

Slaves finding a higher mode on the bus than they are capable of following return to the monitoring state.

5.3 Master field

If two or more masters are transmitting in the same mode arbitration continues into the master field. Due to the wired-AND property, the master with the lowest address will win the arbitration. Masters put their address on the bus with the most significant bit (MSB) first. After each bit they read and compare. If the bus condition differs from their address bit, they relinquish their attempt for the line and continue as potential slaves. At the end of the address field of 12 bits only 1 master is still transmitting. This master then closes the address field with a parity bit enabling other devices to check the validity of the address.

5.4 Slave field

Having won the arbitration, the master transmits the address of the unit it requires as a slave. The most significant bit is sent first. As mentioned in section 5.3, each unit's address consists of 12 bits. After the address of the slave unit, a parity bit is sent to test whether the address is accurately received.

To find out whether or not the slave unit is listening on the bus, the master asks for an acknowledge. All possible slave units are obliged to read the bus at least until the slave address bits do not agree with their own address. When this happens, the unit returns to the monitoring state.

If the slave address matches the address of one of the listening units, and the parity of both master and slave addresses is found to be odd (and therefore correct), the slave unit generates an acknowledge. If the parity is even, the addressed slave does not give an acknowledge - indicating the incorrect reception of the address. It then returns to the monitoring state.

5.5 Control field

In this field the master puts a 4-bit word on the bus (MSB first). This word describes the nature of the message (e.g., direction of transfer, whether the slave should be locked to the master, or whether the data is actual data, a memory address or a command). The control bits are more fully described in section 7.1. The control field is closed with a parity bit followed, as in the slave address field, with an acknowledge bit. If the slave acknowledges, the master proceeds to the next field, if not the master may try to repeat the transfer.

The slave reads the control bits and checks the parity. If it cannot perform the function or the parity is even, it does not acknowledge the master and returns to the monitoring state. If the parity is odd and it can execute the function required by the master it gives an acknowledge and proceeds to the next field.

5.6 Data field

In this field, the master writes data to or reads data from the slave (MSB first). Each byte consists of 8 bits followed by an end-of-data (EOD) bit, a parity bit and an acknowledge bit.

Archief
Ir. A.P.M.Moelands

The end-of-data bit informs the receiver whether or not the present byte is the last one of the message. This bit is necessary because in each mode a message does not have to fill its allocated time slot.

Thus, if a message contains less than the maximum number of bytes, a logic zero in the end-of-data bit of the last byte indicates that the message is over, provided the following acknowledge bit is received or transmitted correctly. After the end-of-data bit, the parity bit and acknowledge bit are sent.

5.6.1 Write action on the bus

In the case of a write action on the bus, the master transmits the data bits, the end-of-data bit, the parity bit, and requests an acknowledge.

The slave collects the data, the end-of-data bit and the parity bit. The parity is then checked and if correct, the acknowledge is given.

If the parity-sum is even, the byte and EOD bit are rejected and the slave does not acknowledge.

If the byte is rejected and there is still time in the time slot for a retry, the master will send the byte again and continue to do so until the byte is accepted or the time slot has expired.

If the parity-sum is odd, the slave accepts the data and generates an acknowledge. The master then starts transmitting the next data byte, if the message wasn't finished or the time slot hadn't expired.

The end-of-data bit is generated by the master-transmitter. This value is a logic '1' when the master still has another byte to transmit and the present byte-number is not the maximum number that the message can contain. The end-of-data bit is a logic '0' when the master has no more data to send, or when the present byte number is the maximum number that the message can contain.

5.6.2 Read action on the bus

For a read action, the master sends all the synchronization edges and the acknowledge bit. During the data periods of the data bits, the end-of-data bit and the parity bit, the slave puts the required levels on the bus.

The master collects the data, the end-of-data bit and the parity bit; then the parity-sum is checked.

If it is even, the byte and EOD bit are rejected and the master-receiver doesn't acknowledge. If the byte is rejected and there is still time in the time slot then the master will read the byte again and continue to do so until either the byte is accepted or the time slot expires.

If the parity-sum is odd, the master accepts the data and generates an acknowledge.

If this byte is not accompanied by an EOD '0' and the time slot has not expired, the master reads another byte.

6.0 BIT PROTOCOL

In this section the state table given in italics details the exact state of the bus during each section of each bit. This information is not required in the first reading and may be omitted.

The bits placed upon the D²B comprise four sections in the time domain (Fig. 6.1):

- an initial HIGH period at logic level '1', the PREPARATION PERIOD.
- a following LOW period at logic level '0', the SYNCHRONIZATION PERIOD.
- a period containing the actual bit value, the DATA PERIOD.
- a final period at logic level '1', the STOP PERIOD.

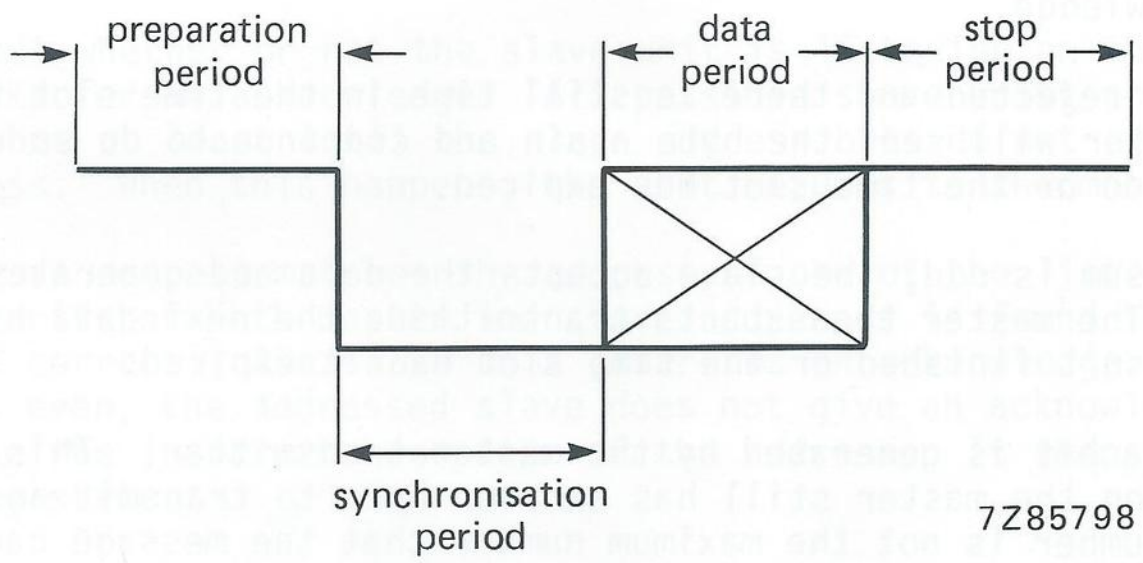


Fig. 6.1 The general bit format on the D²B.

The duration of the bits and their subdivision depend on the mode in use, the type of bit being transmitted, the specified maximum propagation times through the bus and the maximum fall and rise times of the signal at a receiving port (see section 9.0 for more detailed timing requirements).

In the case of bits taking part in arbitration, the second and third parts of the bits are divided into two time intervals. This is to permit an arbitrating master to check the conditions on the bus before proceeding.

In the following definitions of the various types of data bits, the time intervals relating to the master are specified as T1, T2 etc. Time intervals relating to the slave are specified as Ta, Tb etc. (see section 9.1)

Note: Where the state or counter content has an " ' " indication, this state is part of the next bit.

The calculation of the timing parameters has taken into account the delay caused by the data-acquisition circuit. In the reference circuit shown (Fig. 6.2), the first flip-flop (FF_0) synchronizes the asynchronous bus signal with the filter clock. Thereafter the other two flip-flops (FF_1 and FF_2) ensure that the signal Q2 remains stable for at least two clock cycles. Flip-flop FF_3 synchronizes the asynchronous filter output signal (Q2) with the logic clock.

The clocks of the filter and the logic may be asynchronous but their frequencies must fulfill the stated requirements for the clocks in each mode.

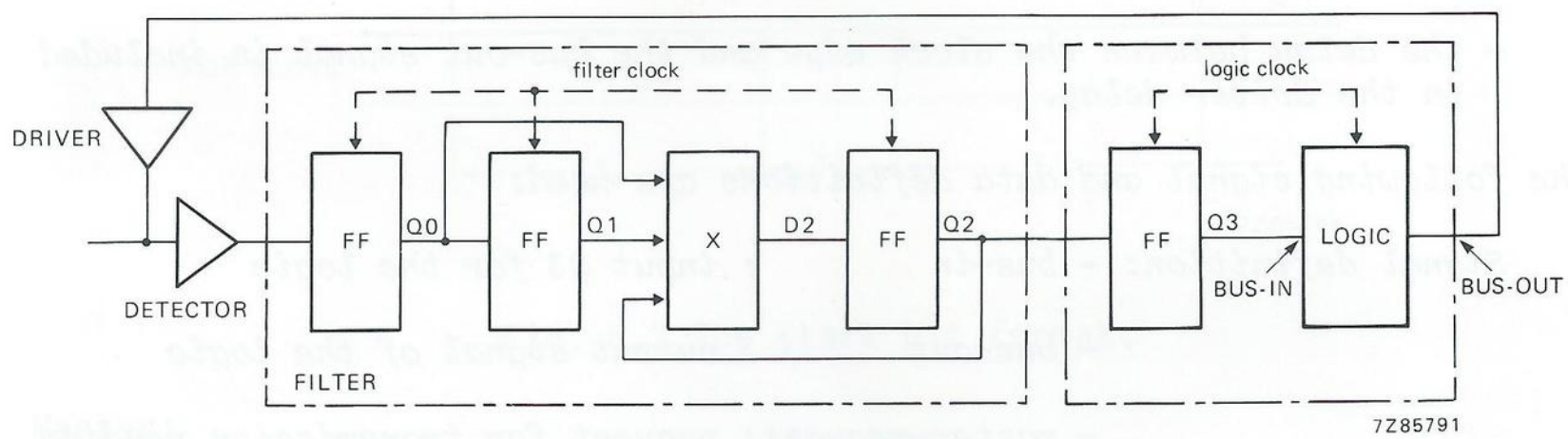


Fig. 6.2 Reference data-acquisition circuit.

Logic function X

Q0	Q1	Q2	D2
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

6.1 Bit formats

The following construction is used to describe the sequence of states in the various bit types:

STATE (X) : *a certain time section X of a bit*

CONDITION : *the conditions valid on the clock edge*

ACTION : *actions to be executed on the clock edge on which the conditions are true*

The following constraints are imposed on the logic handling the bit timing:

- *the result of the actions and the ensuing conditions must be stable within one clock period.*
- *the delay between the clock edge and the bus-out signal is included in the driver delay.*

The following signal and data definitions are used:

Signal definition:

- *bus-in* : *input Q3 for the logic*
- *bus-out* : *output signal of the logic*
- *master-request*: *request for transmission pending*
- *relevant-bit* : *represents the value ('0' or '1') of the bit that is transmitted*
- *bit value* : *represents the value ('0' or '1') of the bit that is received*
- *arbitration-not-lost*
- *arbitration-lost*
- *last-bit-in-the-message*
- *not-last-bit-in-the-message*
- *timing error* : *timing error handling and after the timing error handling go to STATE (Ta) of start bit*

Data definition:

- *clcnt* : *counter which counts clock pulses*

6.1.1 Start bit

The unit wanting to be bus master tests the bus for a HIGH level. If a LOW is detected, it indicates that the bus is already in use. If the bus stays HIGH during this time, the master sends the start bit. The start bit is a signal to all other units that a transmission is beginning.

Fig. 6.3 illustrates the start bit format.

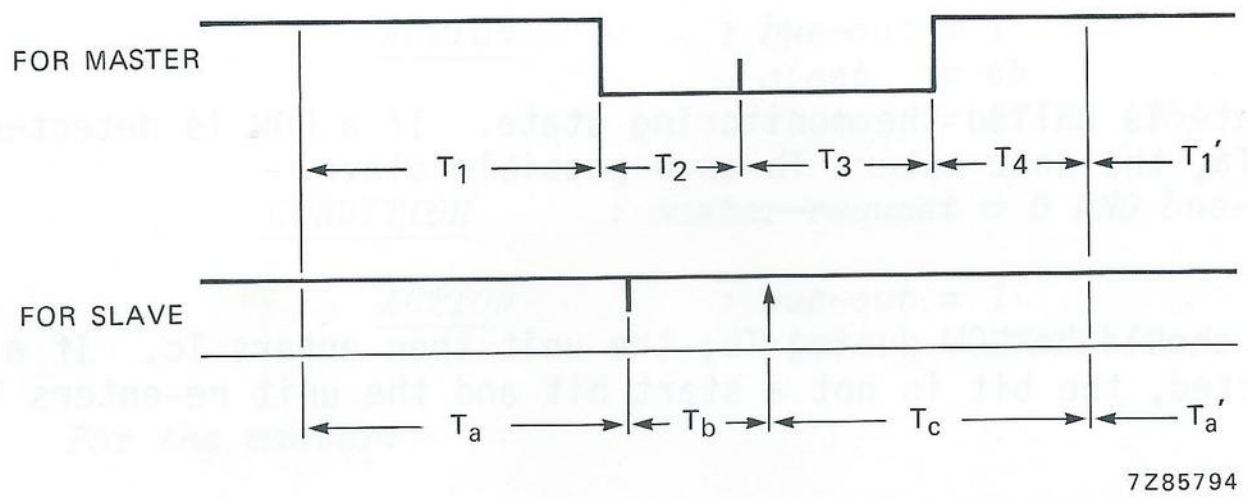


Fig. 6.3 The start bit format.

Master:

- T1:

  If a unit wishes to become master and the bus is HIGH during T1, it enters T2. The HIGH indicates that no other unit has either generated a start condition, or is busy on the bus.

  If a unit wishes to become bus master during T1 and a LOW is detected, the unit becomes arbitrating-master and enters Tb (slave timing) keeping its output HIGH.

Note: The reason an arbitrating-master can perform slave timing is so that it does not disturb a message in progress. However, the other units on the bus - whether arbitrating-masters or slaves - must be able to recognize the start bit. The only way for them to do this, is by following Tb and Tc to detect the long LOW period on the bus which indicates the generation of the start bit.

- T2:

  When a master enters T2, it puts a LOW on the bus and samples the bus for this LOW. When this LOW is found on the bus, the master enters T3. If no LOW is found, a timing error has occurred.

- T3:

  The bus should be kept LOW during T3, the master then enters T4 and generates a HIGH on the bus. If a HIGH is detected during T3, a timing error has occurred - since T3 is the interval which indicates the unique long LOW period of the start bit.

- T4:

  When the HIGH is detected during T4, the master enters T1'. If the bus is still LOW at the end of T4, a timing error has occurred.

Slave:

- Ta:

  This state is called the monitoring state. If a LOW is detected during Ta, the unit enters Tb as a possible slave.

- Tb:

  The bus should be LOW during Tb; the unit then enters Tc. If a HIGH is detected, the bit is not a start bit and the unit re-enters Ta.

- Tc:

  Although a unit can be performing the slave timing during the start bit, it is still possible for it to become a master. If during Tc, a HIGH is detected and the unit now wishes to become the master, it enters T1' and follows the master timing.

  If a unit does not want to be a master, it enters Ta' as a possible slave and follows slave timing

  If the bus is still LOW at the end of Tc, a timing error has occurred.

For specific bit timings see section 9.1

The state sequence for the start bit is as follows:

Initial:

STATE (Ta)

CONDITION : *master-request = 1 AND bus-in = 1*

ACTION : *bus-out = 1*
clcnt = t1
state = STATE (T1)

CONDITION : *bus-in = 0*

ACTION : *bus-out = 1*
clcnt = tb
state = STATE (Tb)

CONDITION : *master-request = 0 AND bus-in = 1*

ACTION : *bus-out = 1*
state = STATE (Ta)

For the master:

STATE (T1)

CONDITION : *bus-in = 0*

ACTION : *bus-out = 1*
clcnt = tb
state = STATE (Tb)

CONDITION : *bus-in = 1 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T1)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 0*
clcnt = t2
state = STATE (T2)

Archief
Ir. A.P.M.Moelands

STATE (T2)

CONDITION : *bus-in = 0*

ACTION : *bus-out = 0*
clcnt = t3
state = STATE (T3)

CONDITION : *bus-in = 1 AND clcnt > 1*

ACTION : *bus-out = 0*
clcnt = clcnt - 1
state = STATE (T2)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

STATE (T3)

CONDITION : *bus-in = 0 AND clcnt > 1*

ACTION : *bus-out = 0*
clcnt = clcnt - 1
state = STATE (T3)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = t4
state = STATE (T4)

CONDITION : *bus-in = 1*

ACTION : *bus-out = 1*
timing error

STATE (T4)

CONDITION : *bus-in = 0 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T4)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

CONDITION : *bus-in = 1*

ACTION : *bus-out = 1*
clcnt = t1'
state = STATE (T1')

For the slave:

STATE (Tb)

CONDITION : *bus-in = 0 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Tb)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = tc
state = STATE (Tc)

CONDITION : *master-request = 0 AND bus-in = 1*

ACTION : *bus-out = 1*
state = STATE (Ta)

CONDITION : *master-request = 1 AND bus-in = 1*

ACTION : *bus-out = 1*
clcnt = t1
state = STATE (T1)

STATE (Tc)

CONDITION : *bus-in = 0 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Tc)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

CONDITION : *master-request = 0 AND bus-in = 1*

ACTION : *bus-out = 1*
clcnt = ta'
state = STATE (Ta')

CONDITION : *master-request = 1 AND bus-in = 1*

ACTION : *bus-out = 1*
clcnt = t1'
state = STATE (T1')

6.1.2 Arbitration bit

For arbitration, which takes place during the mode and master address bits, the bit format is shown in Fig. 6.4.

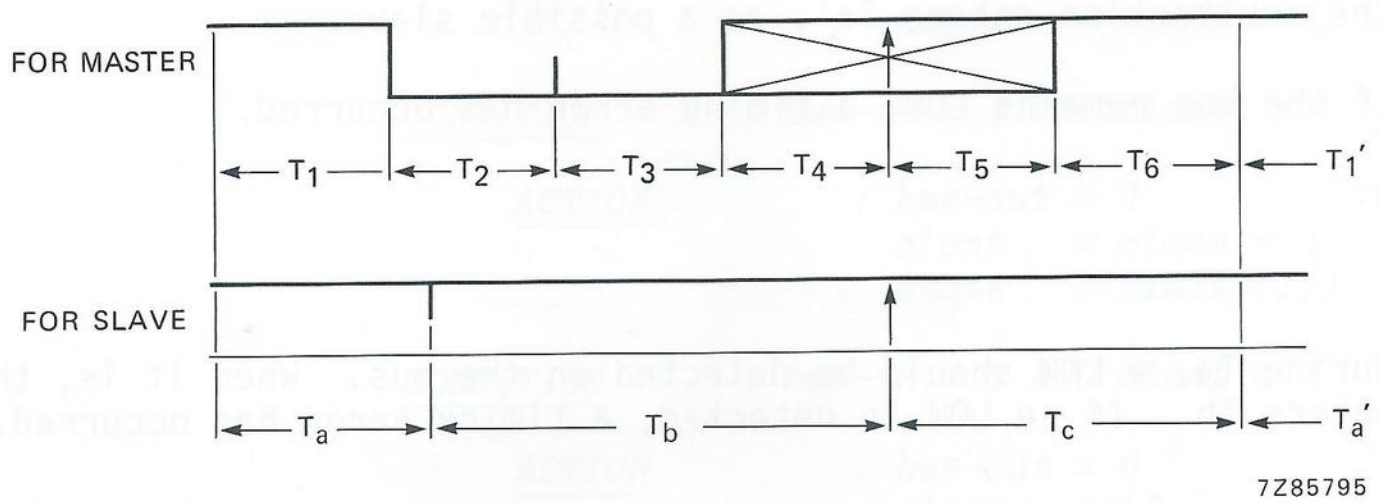


Fig. 6.4 Arbitration bit format.

- T1:

  If the bus remains HIGH during T1, the unit enters T2. If a LOW is detected, a master enters T3, keeping its output HIGH. This indicates that another master has generated the synchronization pulse for this bit.

- T2:

  A master has to generate the synchronization pulse by pulling the bus LOW. As soon as this LOW is found on the bus, the master enters T3 keeping its output LOW. If the LOW is not found, a timing error has occurred.

- T3:

  At the end of T3, the master enters T4.

- T4:

  At the beginning of T4, the arbitrating-master transmits the relevant bit for arbitrating with other units. At the end of T4, the unit reads the bus and enters T5.

  When the bus level is the same as its bit level, the master continues arbitrating. If the levels are different (i.e. there is a LOW on the bus when a HIGH was generated), it has lost the arbitration.

- T5:

  If the bus is LOW at the end of T5, the unit enters T6. If the bus is HIGH, the next state is entered (T1'). A master which has lost the arbitration enters Ta' instead of T1'.

- T6:

  The master generates a HIGH and when this is detected during T6, the arbitrating-master enters the next state (T1'). A unit which has lost the arbitration enters Ta', as a possible slave.

  If the bus remains LOW, a timing error has occurred.

Slave:

- Ta:

  During Ta, a LOW should be detected on the bus. When it is, the unit enters Tb. If no LOW is detected, a timing error has occurred.

- Tb:

  At the end of Tb, the slave samples the bus. When the sample is HIGH, the next state is entered (Ta'). If not, the slave enters Tc.

- Tc:

  When a HIGH is detected during Tc, the next state is entered (Ta'). If the bus remains LOW, a timing error has occurred.

Archief
Ir. A.P.M.Moelands

The state sequence for the arbitration bit is :

For the master:

STATE (T1)

CONDITION : *bus-in = 0*

ACTION : *bus-out = 1*
clcnt = t3
state = STATE (T3)

CONDITION : *bus-in = 1 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T1)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 0*
clcnt = t2
state = STATE (T2)

STATE (T2)

CONDITION : *bus-in = 0*

ACTION : *bus-out = 0*
clcnt = t3
state = STATE (T3)

CONDITION : *bus-in = 1 AND clcnt > 1*

ACTION : *bus-out = 0*
clcnt = clcnt - 1
state = STATE (T2)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

STATE (T3)

CONDITION : *bus-out = 0 AND clcnt > 1*

ACTION : *bus-out = 0*
clcnt = clcnt - 1
state = STATE (T3)

CONDITION : *bus-out = 1 and clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T3)

CONDITION : *clcnt = 1*

ACTION : *bus-out = relevant bit*
clcnt = t4
state = STATE (T4)

STATE (T4)

CONDITION : *clcnt > 1*

ACTION : *bus-out = relevant bit*
clcnt = clcnt - 1
state = STATE (T4)

CONDITION : *bus-in = 0 AND clcnt = 1 AND relevant bit = 0*

ACTION : *bus-out = relevant bit*
clcnt = t5
bit value = 0
arbitration-not-lost
state = STATE (T5)

CONDITION : *bus-in = 1 AND clcnt = 1 AND relevant bit = 0*

ACTION : *bus-out = 1*
timing error

CONDITION : *bus-in = 0 AND clcnt = 1 AND relevant bit = 1*

ACTION : *bus-out = relevant bit*
clcnt = t5
bit value = 0
arbitration-lost
state = STATE (T5)

CONDITION : *bus-in = 1 AND clcnt = 1 AND relevant bit = 1*

ACTION : *bus-out = relevant bit*
clcnt = t5
bit value = 1
arbitration-not-lost
state = STATE (T5)

Archief
Ir. A.P.M.Moelands

STATE (T5)

CONDITION : *clcnt > 1*

ACTION : *bus-out = relevant bit*
clcnt = clcnt - 1
state = STATE (T5)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = t6
state = STATE (T6)

CONDITION : *bus-in = 1 AND clcnt = 1 AND arbitration-not-lost*

ACTION : *bus-out = 1*
clcnt = t1'
state = STATE (T1')

CONDITION : *bus-in = 1 AND clcnt = 1 AND arbitration-lost*

ACTION : *bus-out = 1*
clcnt = ta'
state = STATE (Ta')

STATE (T6)

CONDITION : *bus-in = 0 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T6)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

CONDITION : *bus-in = 1 AND arbitration-not-lost*

ACTION : *bus-out = 1*
clcnt = t1'
state = STATE (T1')

CONDITION : *bus-in = 1 AND arbitration-lost*

ACTION : *bus-out = 1*
clcnt = ta'
state = STATE (Ta')

For the slave:

STATE (Ta)

CONDITION : *bus-in = 0*

ACTION : *bus-out = 1*
clcnt = tb
state = STATE (Tb)

CONDITION : *bus-in = 1 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Ta)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

STATE (Tb)

CONDITION : *clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Tb)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = tc
bit value = 0
state = STATE (Tc)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = ta'
bit value = 1
state = STATE (Ta')

STATE (Tc)

CONDITION : *bus-in = 0 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Tc)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

CONDITION : *bus-in = 1*

ACTION : *bus-out = 1*
clcnt = ta'
state = STATE (Ta')

6.1.3 Master to slave bit

Master to slave bits form the address of the unit required as a slave; the control bits which indicate the type of transfer; and the actual data bits (Fig. 6.5). Note that if the control bits stipulate a slave to master transfer, the data bits will follow the format in section 6.1.4.

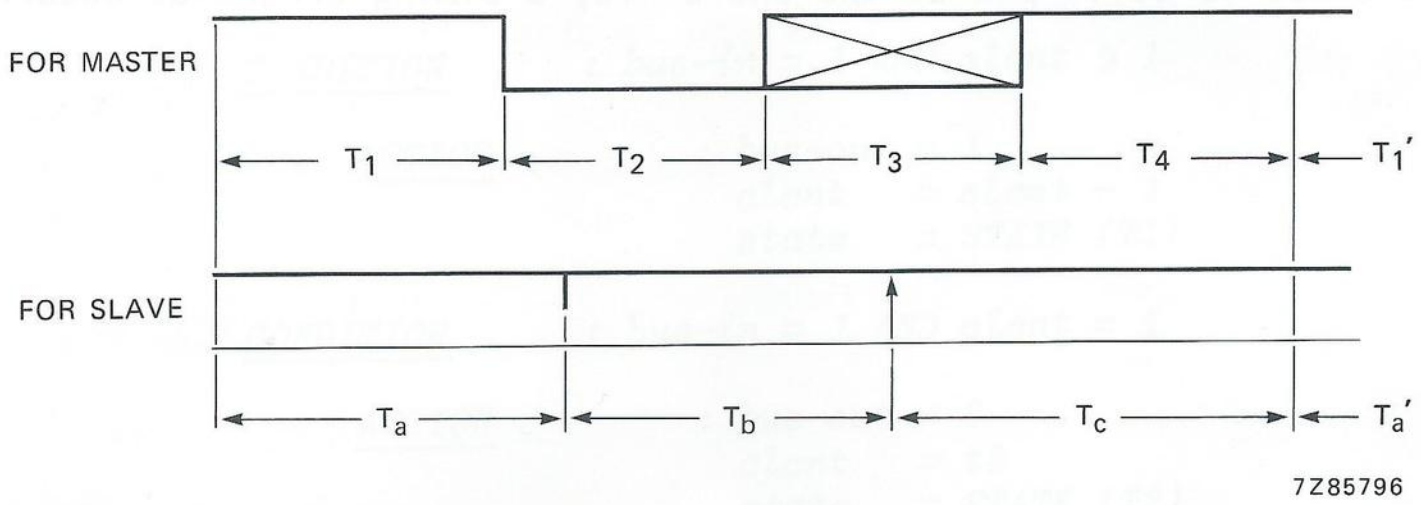


Fig. 6.5 Master to slave bit format.

Master:

- T1:

  The bus should remain HIGH during T1. If it does, the master enters T2. If a LOW is detected, a timing error has occurred.

- T2:

  The master generates the synchronization pulse by pulling the bus LOW. If the bus remains HIGH during T2, a timing error has occurred. If not, the master enters T3.

- T3:

  During T3, the master transmits the relevant data on the bus. It then enters T4.

- T4:

  During T4, the master transmits a HIGH on the bus. If a LOW is detected after this period, an error has occurred. If not, the next state is entered (T2').

Slave:

- Ta:

  A LOW should be detected during Ta. When it is, the slave enters Tb. If the bus remains HIGH, a timing error has occurred.

- Tb:

  At the end of Tb, the unit samples the bus to read the bit. If the bus is HIGH, the next state is entered (Ta'). If it is LOW, the unit enters Tc.

- Tc:

  As soon as a HIGH is detected, the next state is entered (Ta'). If the bus is still LOW at the end of Tc, a timing error has occurred.

Archief
Ir. A.P.M.Moelands

The state sequence for master to slave bits is :

For the master:

STATE (T1)

CONDITION : *bus-in = 0*

ACTION : *bus-out = 1*
timing error

CONDITION : *bus-in = 1 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T1)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 0*
clcnt = t2
state = STATE (T2)

STATE (T2)

CONDITION : *clcnt > 1*

ACTION : *bus-out = 0*
clcnt = clcnt - 1
state = STATE (T2)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = relevant bit*
clcnt = t3
state = STATE (T3)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

STATE (T3)

CONDITION : *clcnt > 1*

ACTION : *bus-out = relevant bit*
clcnt = clcnt - 1
state = STATE (T3)

CONDITION : *clcnt = 1*

ACTION : *bus-out = 1*
clcnt = t4
state = STATE (T4)

STATE (T4)

CONDITION : *clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T4)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

CONDITION : *bus-in = 1 AND clcnt = 1 AND last-bit-in-the-message*

ACTION : *bus-out = 1*
state = STATE (Ta of the start bit)

CONDITION : *bus-in = 1 AND clcnt = 1 AND not-last-bit-in-the-message*

ACTION : *bus-out = 0*
clcnt = t2'
state = STATE (T2')

For the slave:

STATE (Ta)

CONDITION : *bus-in = 0*

ACTION : *bus-out = 1*
clcnt = tb
state = STATE (Tb)

CONDITION : *bus-in = 1 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Ta)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

STATE (Tb)

CONDITION : *clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Tb)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = tc
bit value = 0
state = STATE (Tc)

CONDITION : *bus-in = 1 AND clcnt = 1 AND last-bit-in-message*

ACTION : *bus-out = 1*
bit value = 1
state = STATE (Ta of the start bit)

CONDITION : *bus-in = 1 AND clcnt = 1 AND not-the-last-bit-in-the-message*

ACTION : *bus-out = 1*
clcnt = ta'
bit value = 1
state = STATE (Ta')

STATE (Tc)

CONDITION : *bus-in = 0 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Tc)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

CONDITION : *bus-in = 1 AND last-bit-in-the-message*

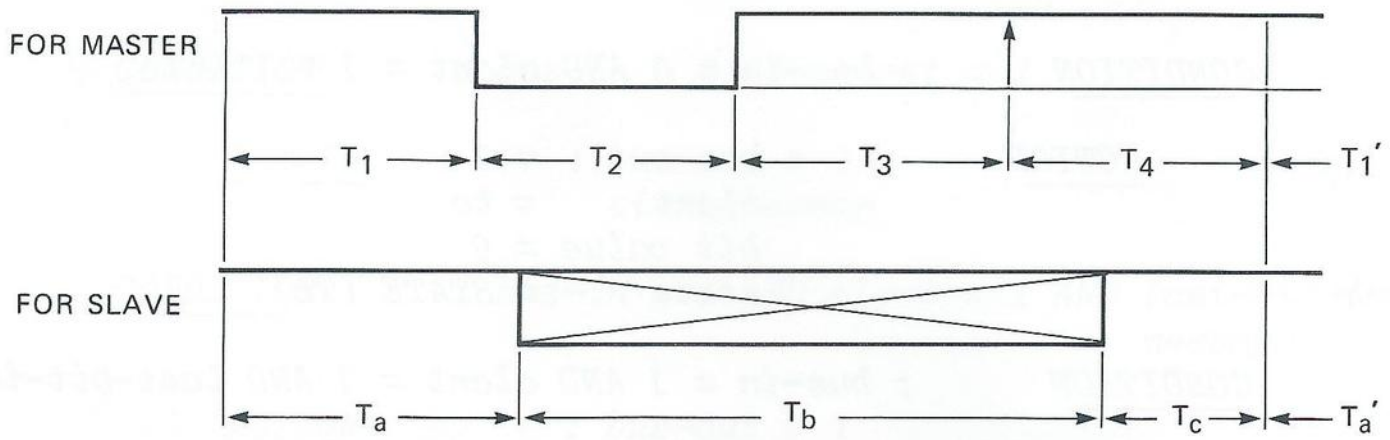
ACTION : *bus-out = 1*
state = STATE (Ta of the start bit)

CONDITION : *bus-in = 1 AND not-last-bit-in-the-message.*

ACTION : *bus-out = 1*
clcnt = ta'
state = (Ta')

6.1.4 Slave to master bit

The slave to master bits form the "slave transmitter to master receiver" data transfer and acknowledge.



7Z85797

Fig. 6.6 Slave to master bit format.

Master:

- T1:

  Does not apply (T1 is always zero).

- T2:

  The master transmits the synchronization pulse by pulling the bus LOW during T2. At the end of this period, when the LOW is detected on the bus, the master enters T3. If the bus is still HIGH after T2, a timing error has occurred.

- T3:

  During T3, the master generates a HIGH on the bus. After t3, it samples the bus to read the bit transmitted by the slave and enters T4.

- T4:

  The bus should be HIGH at the end of T4. The master then enters the next state (T2'). If the bus is still LOW, a timing error has occurred.

Slave:

- Ta:

  When the LOW generated by the master is detected, the slave enters Tb. If the bus is still HIGH at the end of Ta, a timing error has occurred.

Archief
Ir. A.P.M.Moelands

- Tb:

  During Tb, the slave transmits the relevant bit. If at the end of Tb the bus is HIGH, the slave enters (Ta'). If the bus is LOW, the slave enters Tc.

- Tc:

  During Tc, a HIGH should be detected on the bus. When it is, the next state is entered (Ta'). If the bus remains LOW, a timing error has occurred.

The state sequence for slave to master bits is:

For the master:

STATE (T1)

(Does not apply)

STATE (T2)

CONDITION : *clcnt > 1*

ACTION : *bus-out = 0*
clcnt = clcnt - 1
state = STATE (T2)

CONDITION : *bus-in = 0 AND clcnt =1*

ACTION : *bus-out = 1*
clcnt = t3
state = STATE (T3)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

STATE (T3)

CONDITION : *clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T3)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = t4
bit value = 0
state = STATE (T4)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = t4
bit value = 1
state = STATE (T4)

STATE (T4)

CONDITION : *clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (T4)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 0*
timing error

CONDITION : *bus-in = 1 AND clcnt = 1 AND last-bit-in-the-message*

ACTION : *bus-out = 1*
state = STATE (Ta of the start bit)

CONDITION : *bus-in = 1, clcnt = 1 AND not-last-bit-in-the-message*

ACTION : *bus-out = 0*
clcnt = t2'
state = STATE (T2')

For the slave:

STATE (Ta)

CONDITION : *bus-in = 0*

ACTION : *bus-out = relevant bit*
clcnt = tb
state = STATE (Tb)

CONDITION : *bus-in = 1 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Ta)

CONDITION : *bus-in = 1 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

STATE (Tb)

CONDITION : *clcnt > 1*

ACTION : *bus-out = relevant bit*
clcnt = clcnt - 1
state = STATE (Tb)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
clcnt = tc
state = STATE (Tc)

CONDITION : *bus-in = 1 AND clcnt = 1 AND last-bit-in-the-message*

ACTION : *bus-out = 1*
state = STATE (Ta of the start bit)

CONDITION : *bus-in = 1 AND clcnt = 1 AND not-last-bit-in-the-message*

ACTION : *bus-out = 1*
clcnt = ta'
state = STATE (Ta')

STATE (Tc)

CONDITION : *bus-in = 0 AND clcnt > 1*

ACTION : *bus-out = 1*
clcnt = clcnt - 1
state = STATE (Tc)

CONDITION : *bus-in = 0 AND clcnt = 1*

ACTION : *bus-out = 1*
timing error

CONDITION : *bus-in = 1 AND last-bit-in-the-message*

ACTION : *bus-out = 1*
state = STATE (Ta of the start bit)

CONDITION : *bus-in = 1 AND not-last-bit-in-the-message*

ACTION : *bus-out = 1*
clcnt = ta'
state = STATE (Ta')

Archief
Ir. A.P.M.Moelands

7.0 USE OF BITS

In this section a detailed description is given of the meaning of:

- Control bits
- End-of-data bit
- Parity bit
- Acknowledge bit

7.1 Control bits

The 4 control bits define the kind of exchange that has to take place. The following table gives the meaning of the different combinations of the control bits.

B3	B2	B1	B0	FUNCTION
0	0	0	0	Read slave status
0	0	0	1	Reserved for future use
0	0	1	0	Read slave status and lock
0	0	1	1	Read data and lock
0	1	0	0	Read medium and least significant lock address nibble*
0	1	0	1	Read most significant lock address nibble**
0	1	1	0	Read slave status and unlock
0	1	1	1	Read data and unlock
1	0	0	0	Write memory address and lock
1	0	0	1	Reserved for future use
1	0	1	0	Write commands and lock
1	0	1	1	Write data and lock
1	1	0	0	Reserved for future use
1	1	0	1	Reserved for future use
1	1	1	0	Write commands and unlock
1	1	1	1	Write data and unlock

* The least significant lock address nibble is transferred in the least significant part of the byte.

** The most significant lock address nible is transferred in the least significant part of the byte.

Note: When a unit A is locked to a unit B, then:

- unit A has to execute the control functions coded:

B3	B2	B1	B0	
0	0	0	0	- Read slave status
0	1	0	0	- Read medium and least significant lock address nibble
0	1	0	1	- Read high significant lock address nibble

  asked for by units other than unit B.

  Note: the control codes 0000, 0100 and 0101 do not affect the lock status of the slave.

- unit A is allowed to execute all the control functions asked for by unit B.

7.2 End-of-data bit

To indicate that a byte is the last data byte within the current transfer, a bit is added to each data byte.

End-of-data

'1' - Not the last data byte.
'0' - The last data byte in the current transfer.

Note: The end-of-data bit is included in the parity check.

7.3 Parity bit

Parity bits are used to protect:

- master bits
- slave bits
- control bits
- data + end-of-data bit

The parity defined is odd parity. If the bits protected by a parity bit comprise an even number of ones, the parity bit will be a logical 1. If the bits comprise an odd number of ones, the parity bit will be a logical 0.

7.4 Acknowledge bit

During every transfer there are three different acknowledge bits, the first after the slave address, the second after the control bits and the third after each data byte.

Every data byte has to be acknowledged:

'0' indicates a positive acknowledge
'1' indicates a negative acknowledge

The first acknowledge bit should not be transmitted by the addressed slave under the following conditions:

1. Parity fault on the master and/or slave address
2. Mode too high for slave
3. Timing error
4. Slave not present.

In these cases the transmission will be stopped.

The second acknowledge bit should not be transmitted by the slave under the following conditions:

1. Parity fault on the control bits.
2. Receiver buffer is not empty and bit B3 is logic '1'
3. No data in data buffer, the bit B3 is logic '0' and it is not a status or lock address request.
4. The slave is locked to another master, except for status without lock/unlock or a lock-address request.
5. The slave has no memory (see supplement 1) and the master is going to provide a memory address.
6. The receiver is not locked when the lock-address is requested.
7. Timing error
8. Reserved control codes.

In these cases transmission will be stopped.

The third acknowledge bit should not be transmitted under the following conditions:

1. Parity fault seen on the data.
2. Timing error during last data transfer (last = since previous acknowledge).
3. When the receiver buffer is full and it cannot accept another byte.

In these cases, the transmitter has to re-try the byte except when this byte was the last in the time slot.

8.0 ELECTRICAL SPECIFICATION

The D²B cable consists of a differential floating pair forming the data bus (Fig. 8.1).

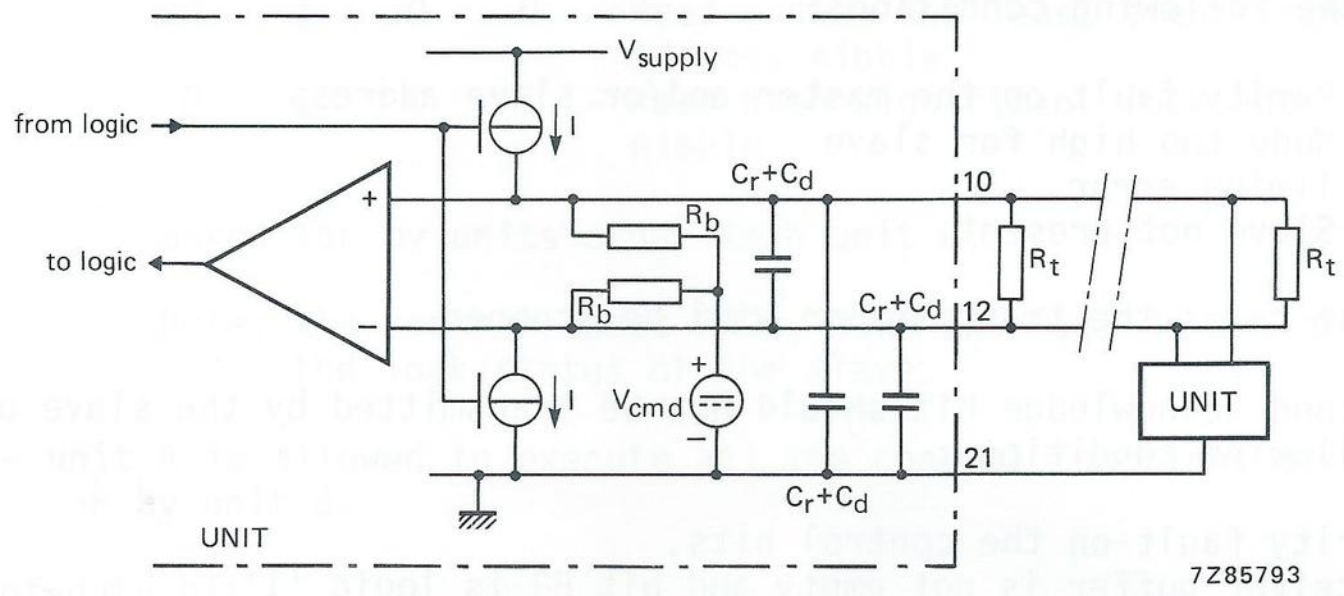


Fig. 8.1 Reference circuit for connecting units to the D²B via a differentially driven cable.

8.1 Logical and electrical state relationship

The relationship between the logical states and the electrical levels on the data lines is given the table below:

Logical state	Electrical levels
0	$\geqslant$ + 120 mV ($V_{10} - V_{12}$) called HIGH
1	$\leqslant$ + 20 mV ($V_{10} - V_{12}$) called LOW

8.2 Driver requirements

The specification for the device drivers is as follows:

- Voltage V_{12} > Voltage V_{21} and V_{10} < 5 V.
- The common mode voltage ($Vcm_d = 0{,}5(V_{10} + V_{12})$ with respect to point 21) is 2,5 V ± 10% under all operating conditions.
- The driver current (I) is:

$$2{,}75\ \text{mA} \leqslant I \leqslant 5\ \text{mA at HIGH}$$
$$I < 1\ \mu\text{A at LOW}$$

- The driver resistance ($R = 2R_b$) between point 10 and 12 must be greater than 100 kΩ for the LOW state.

Archief
Ir. A.P.M.Moelands

- The driver load capacitance (C) is:

C_d < 25 pF between points 10 and 12
C_d < 25 pF between points 10 and 21
C_d < 25 pF between points 12 and 21

- The transition time values (T) are given in Fig. 8.2.

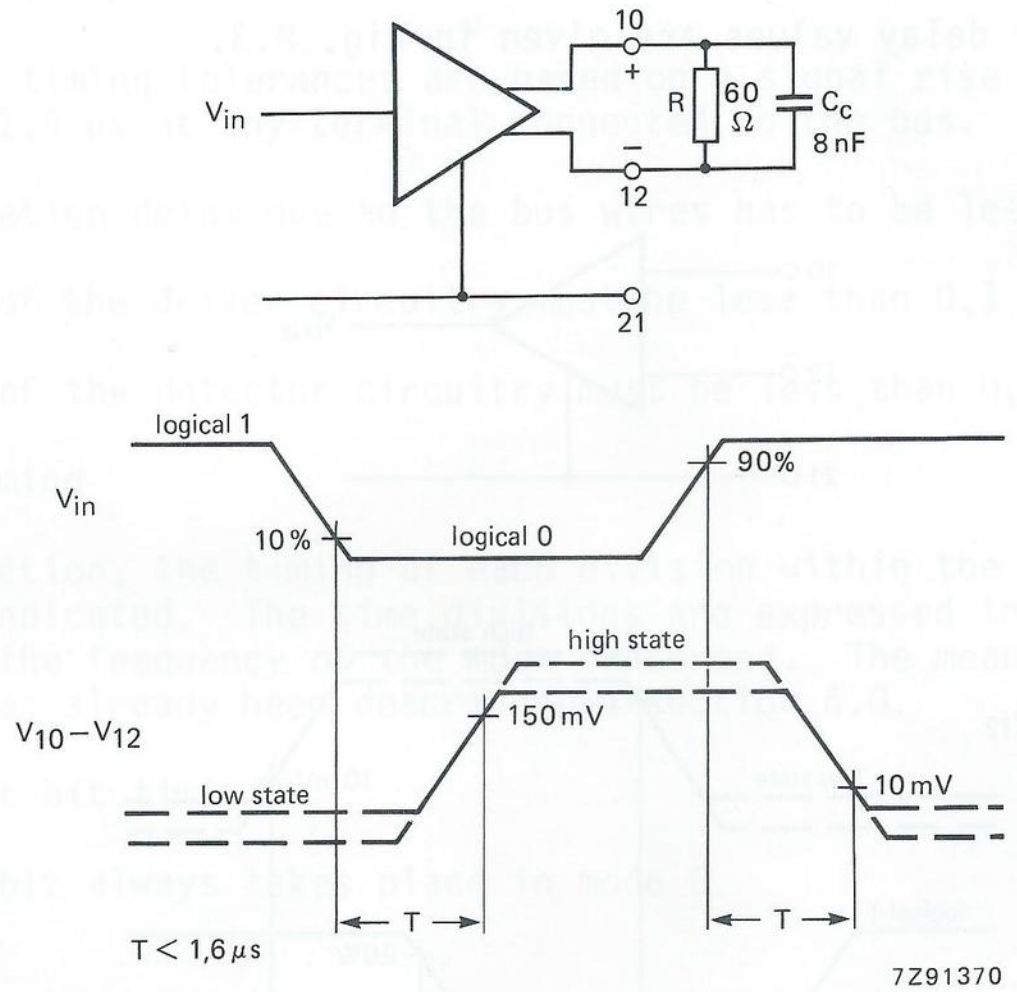


Fig. 8.2 The transition time measurement

8.3 Receiver requirements

The receiver specification is as follows:

- The common mode voltage range ($Vcm_r = 0{,}5(V_{10} + V_{12})$ with respect to point 21) is:

1 V < Vcm_r < 3,75 V for HIGH
0 V < Vcm_r < 5 V for LOW

- The input voltage ($V_{10} - V_{12}$) is:

$V_{10} - V_{12}$ > 120 mV for HIGH
$V_{10} - V_{12}$ < 20 mV for LOW

- Hysteresis must be 20 mV minimum

- The input impedance is:

  $R > 2{,}5\ M\Omega$, $C_r < 25$ pF between points 10 and 21

  $R > 2{,}5\ M\Omega$, $C_r < 25$ pF between points 12 and 21

  $R > 1\ M\Omega$, $C_r < 25$ pF between points 10 and 12

- The receiver delay values are given in Fig. 8.3.

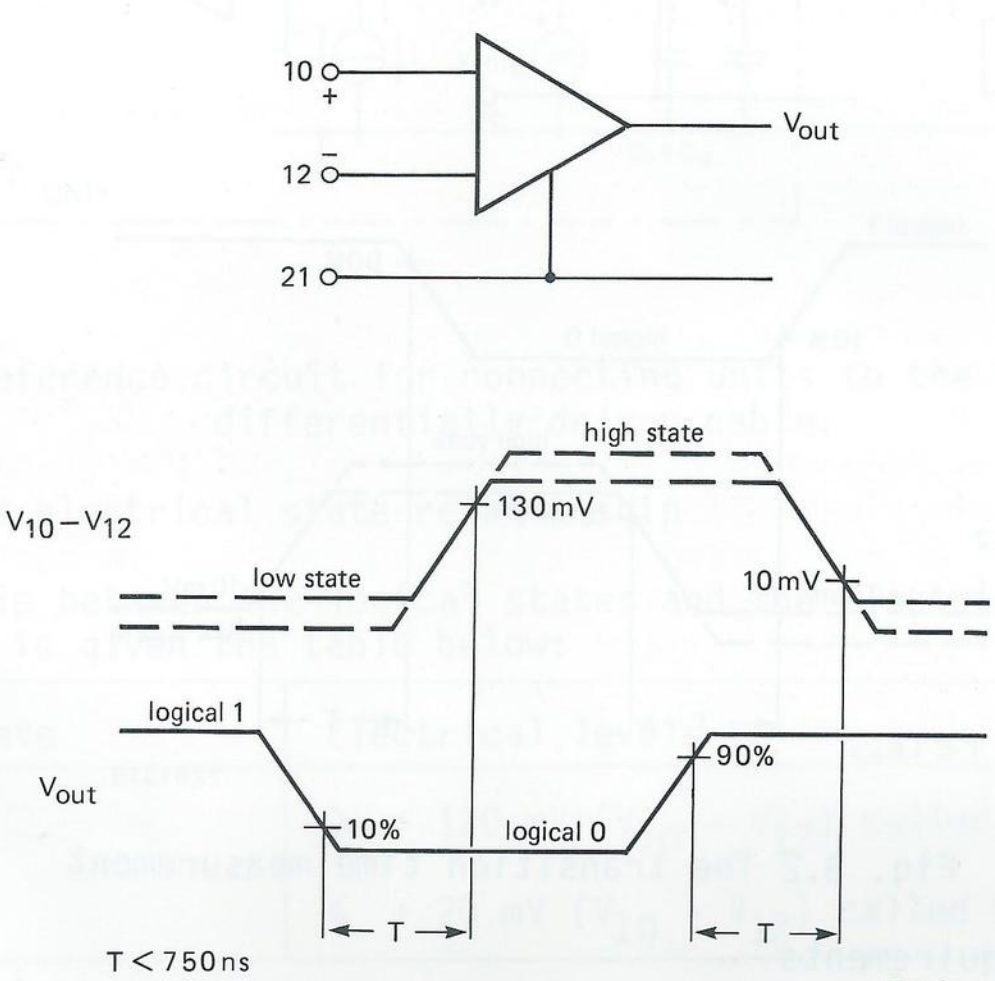


Fig. 8.3 Receiver delay measurement.

8.4 Cable characteristics

- Wire resistance $< 0{,}1\ \Omega/m$
- Characteristic impedance (Z_o) = $120\Omega \pm 20\%$

8.5 Configuration

- Bus length = 150 m (max)
- Resistive cable termination = 120Ω ± 5% at each end
- Up to 50 units can be connected to a well matched bus. If any spurs used, they should be no longer than 2,5 m.

9.0 TIMING

The signal timing tolerances are based on a signal rise and fall time of less than 1,5 µs at any terminal connected to the bus.

The propagation delay due to the bus wires has to be less than 0,9 µs.

The delay of the driver circuitry must be less than 0,1 µs.

The delay of the detector circuitry must be less than 0,75 µs.

9.1 Bit timing

In this section, the timing of each division within the different bit types is indicated. The time divisions are expressed in terms of clock pulses at the frequency of the mode concerned. The meaning of each division has already been described in section 6.0.

9.1.1 Start bit timing

The start bit always takes place in mode 0.

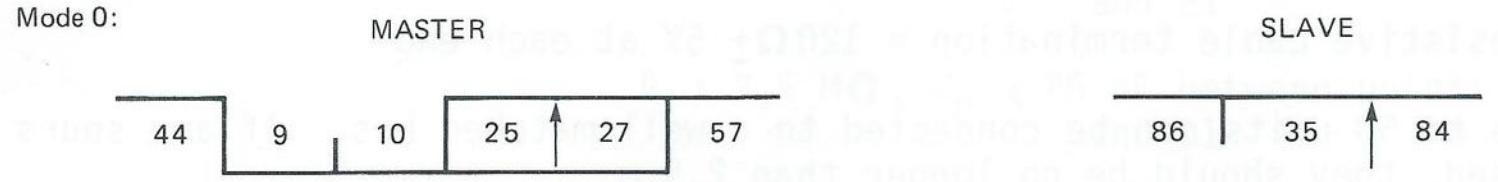


1) the length of this period for a slave is undefined

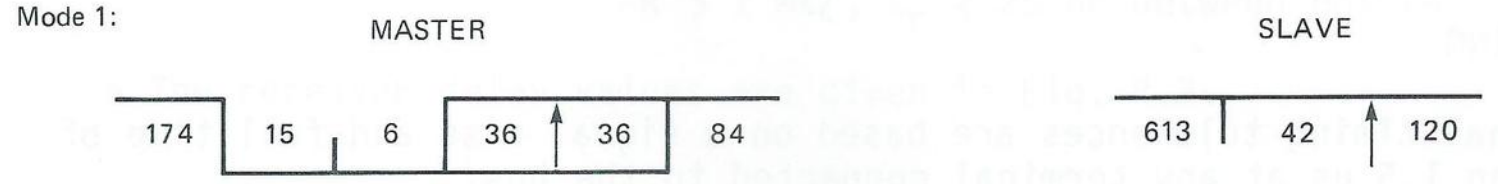
7Z85792

9.1.2 Mode bit timing

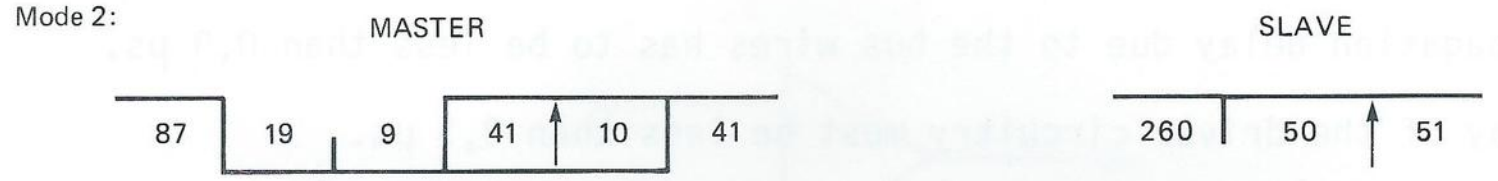
Mode 0:



Mode 1:



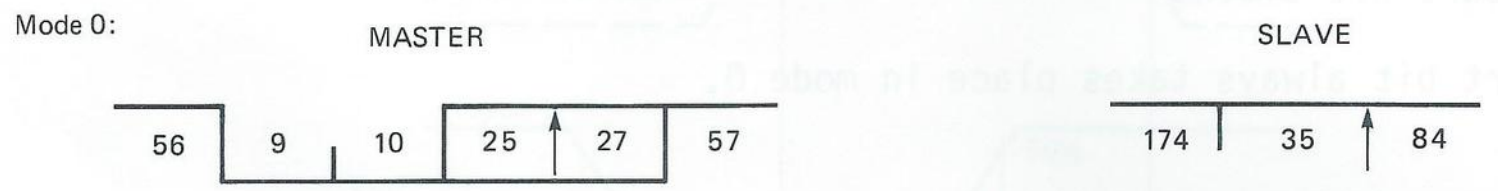
Mode 2:



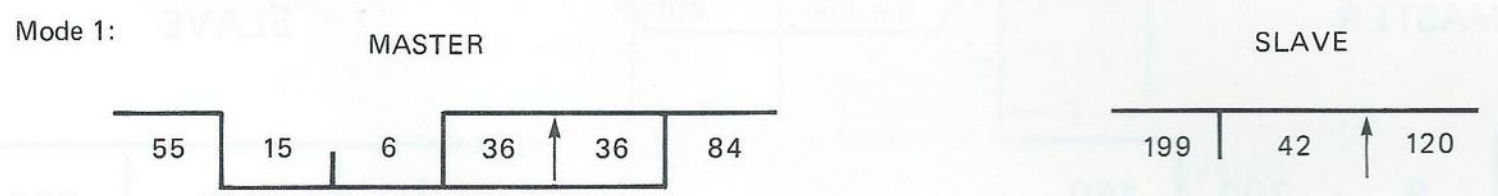
7Z85786

9.1.3 Master address bit timing

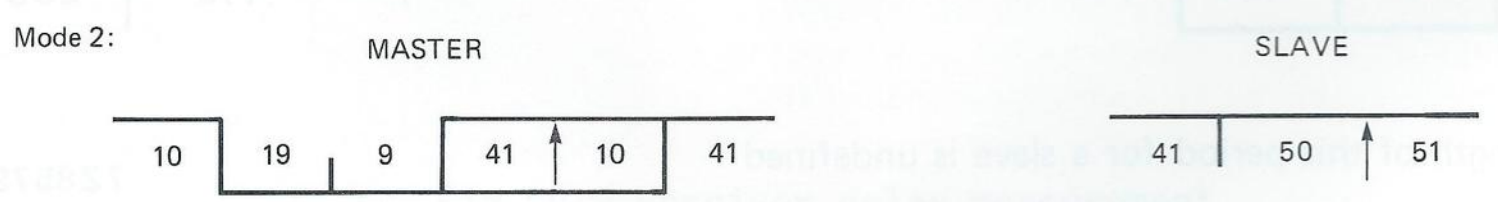
Mode 0:



Mode 1:



Mode 2:

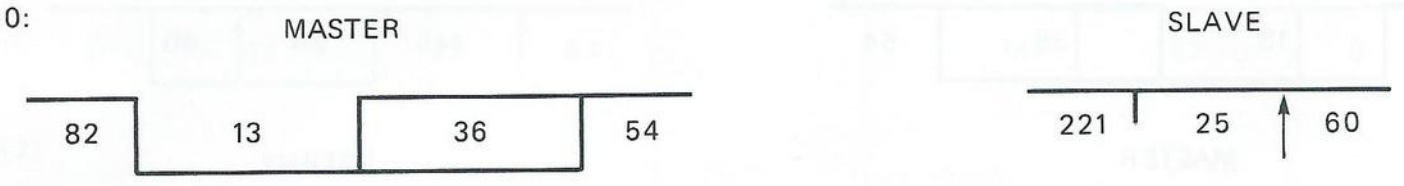


7Z85787

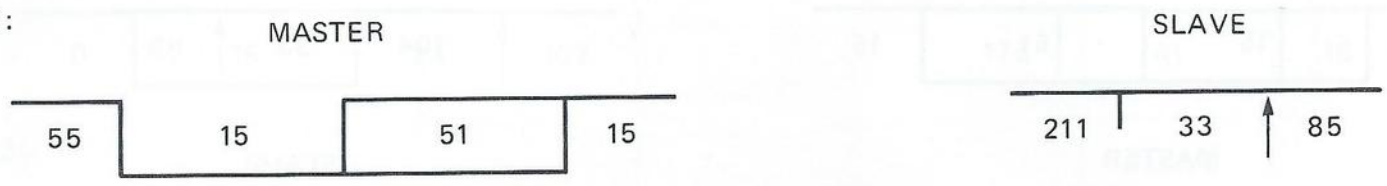
9.1.4 Master to slave bit timing

- Parity bit on master address

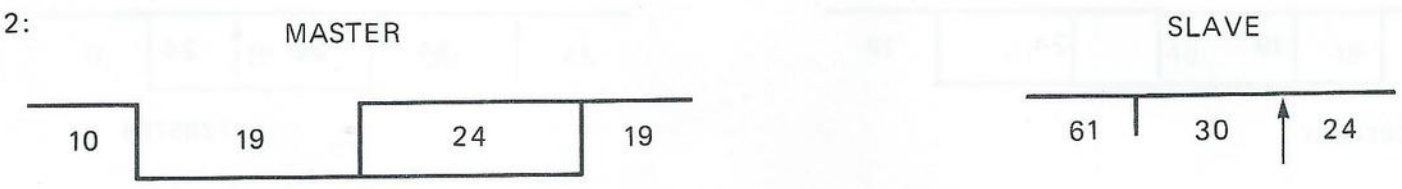
Mode 0:



Mode 1:



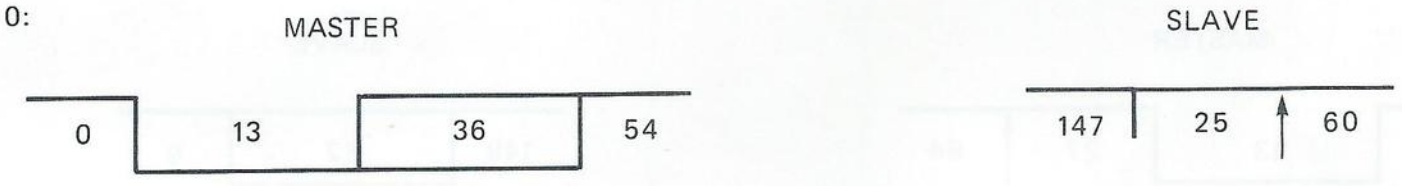
Mode 2:



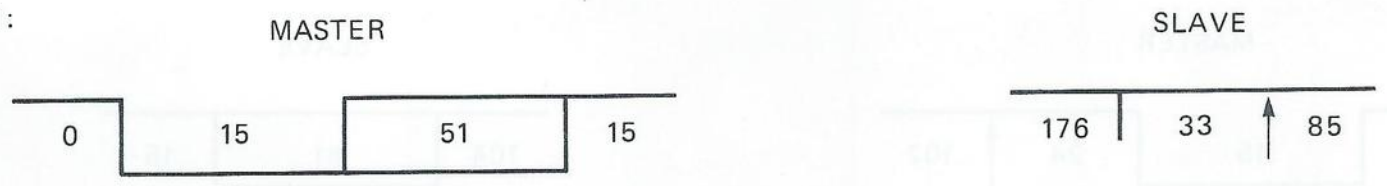
7Z85784

- First bit of the control field (B3),
- First bit of each data byte (master to slave),
- ACK bit, on data in the case of slave to master transfer.

Mode 0:



Mode 1:



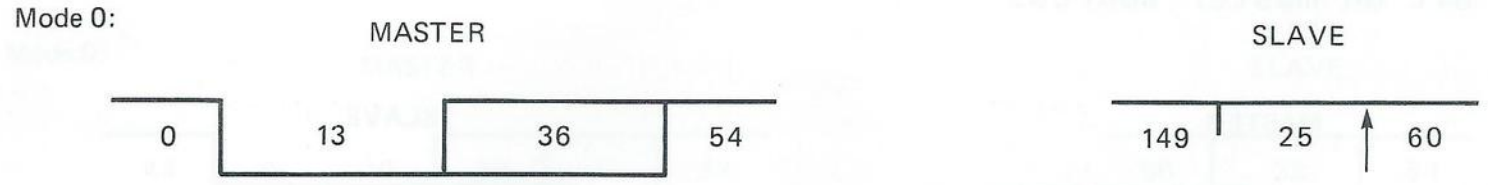
Mode 2:



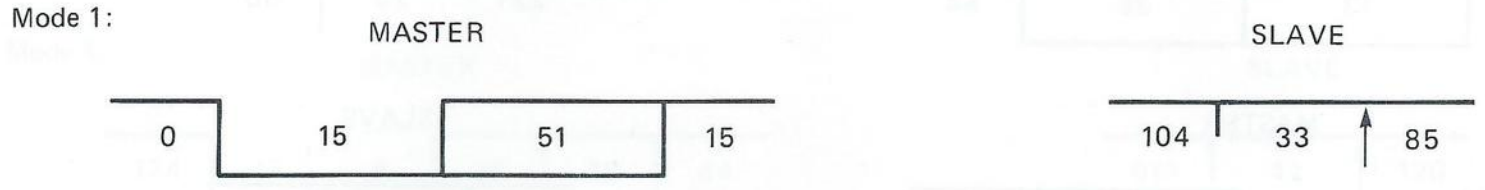
7Z85785

- All other master to slave bits.

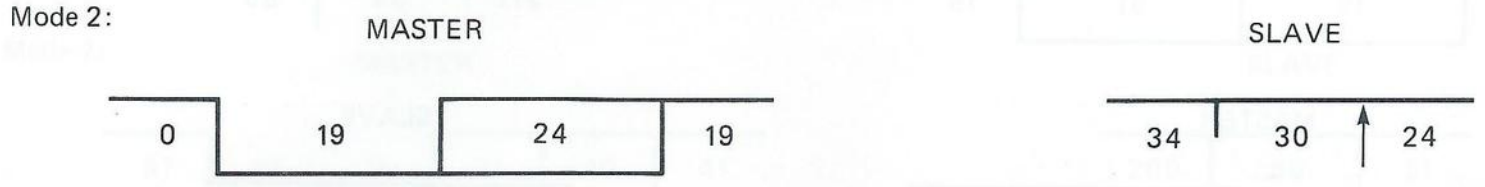
Mode 0:



Mode 1:



Mode 2:



7Z85789

9.1.5 Slave to master bit timing

- ACK bit - slave to master,
- First bit of every data byte (slave to master).
  (except for the first data byte in the message).

Mode 0:



Mode 1:



Mode 2:

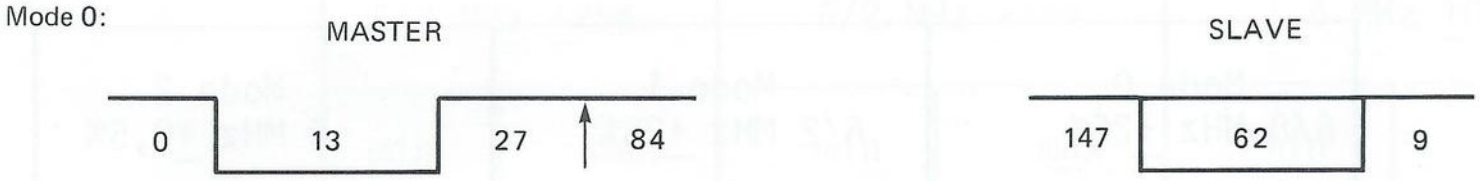


7Z85788

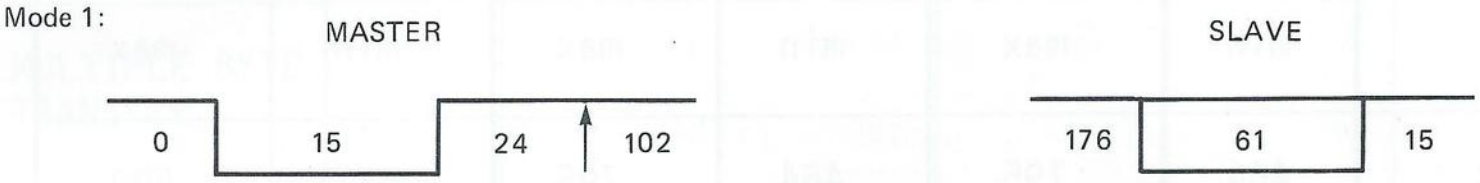
Archief
Ir. A.P.M.Moelands

- All other slave to master bits.

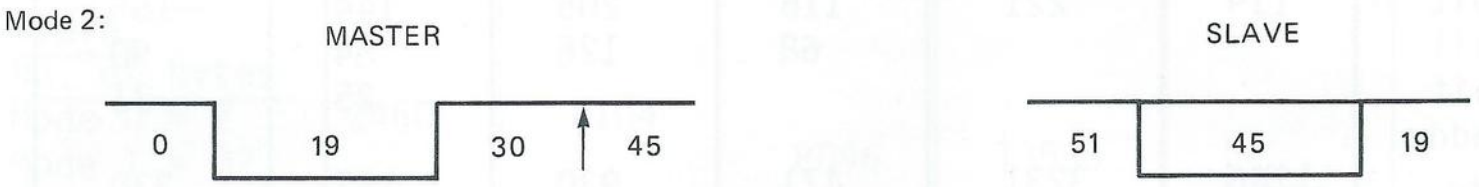
Mode 0:



Mode 1:



Mode 2:



7Z85790

9.2 Message times and transmission speeds (all times given in μs)

	Mode 0 6/8 MHz ±25%		Mode 1 6/2 MHz ±25%		Mode 2 6 MHz ±0,5%	
	min	max	min	max	min	max
Startbit	464	795	464	795	577	599
Mode 0 bit	119	221	116	205	144	155
Mode 1 bit			68	126	84	90
Mode 2 bit					25	31
Master addr. + parity.	1746	3231	471	930	157	230
Slave address + parity + ACK.	1560	2601	318	531	149	151
Control + parity + ACK.	681	1136	145	243	66	68
Overhead	4570	7984	1582	2830	1202	1324
SINGLE BYTE TRANSFER						
Master to slave.						
Data + parity + ACK.	1230	2052	253	423	118	120
Overhead	4570	7984	1582	2830	1202	1324
Total message time.	5800	10036	1835	3253	1320	1444
Transmission speed (c.p.s)	99	173	307	545	692	758
Slave to master.						
Data + parity + ACK.	1432	2388	397	663	166	168
Overhead	4570	7984	1582	2830	1202	1324
Total message time.	6002	10372	1979	3493	1368	1492
Transmission speed (c.p.s)	96	167	286	506	670	731

	Mode 0 6/8 MHz ±25%		Mode 1 6/2 MHz ±25%		Mode 2 6 MHz ±0,5%	
	min	max	min	max	min	max
MULTIPLE BYTE TRANSFER						
Master to slave.						
No. of bytes						
Mode 0 = 2	2460	4104				
Mode 1 = 32			8096	13536		
Mode 2 = 128					15104	15360
Overhead	4570	7984	1582	2830	1202	1324
Total message time.	7030	12088	9678	16366	16306	16684
Transmission speed (c.p.s.)	165	285	1955	3307	7672	7850
Slave to master.						
No. of bytes						
Mode 0 = 2	2864	4776				
Mode 1 = 16			6352	10608		
Mode 2 = 64					10624	10752
Overhead	4570	7984	1582	2830	1202	1324
Total message time.	7434	12760	7934	13438	11826	12076
Transmission speed (c.p.s.)	156	270	1190	2017	5299	5412

SUPPLEMENT 1

1.0 SLAVE STATUS

By requesting the status from a slave it must be possible for a master to determine why the slave doesn't give an acknowledge.

The slave status is only valid for last slave transfer.

Every unit should possess the possibility to provide status information. The status bits have the following meaning:

Bit		Value	Meaning
0	*	0	data buffer empty
		1	data buffer not empty
1	**	0	receiver buffer empty
		1	receiver buffer not empty
2		0	unit not locked
		1	unit locked
3		0	unit has no memory
		1	unit has memory
4		0	slave transmitter section disabled
		1	slave transmitter section enabled
5		0	always set to zero.
7,6		00	mode 0
		01	mode 1 — Indicates the highest mode
		10	mode 2 — in which the unit can work
		11	reserved for future use

(Bit 7 is the MSB bit)

* The data buffer is the buffer which can be accessed by a read data action defined by the control bits.

** The receiver buffer is the buffer which can be accessed by a write action defined by the control bits.

Archief
Ir. A.P.M.Moelands

2.0 LOCK ADDRESS

When the control bits specify "read medium and least significant nibble lock addresses" the interpretation of the 8 data bits (bit 7 is the most significant bit) is:

Bit	Meaning
7	bit 7 of address-locking master

- similarly for bit numbers 6 to 0

The address-locking master contains 12 bits numbered from 0 to 11. Bit number 11 is the most significant bit.

When the control bits specify "read most significant lock address nibble" the interpretation of the 8 data bits (bit 7 is the most significant bit) is:

Bit	Meaning
7-4	undefined
3	bit 11 of address-locking master
2	bit 10 "
1	bit 9 "
0	bit 8 "

3.0 MEMORY ADDRESS

Each unit for audio-video applications has to be implemented with a memory that can be accessed via the control field specification "write memory address".

When a unit receives a message and the control field indicates "write memory address", the received data bytes are processed in the following way.

- the data buffer empty bit in the slave status byte is reset to zero to avoid reading irrelevant data
- the most-significant (MS) bit (bit no. 7) of each byte, starting with the first data byte received, is checked.
  - if the value is zero, the unit will act in the following way:

    the data bits b6 - b0 are interpreted as the 7 MS bits of the memory address and are stored in the MS part of the memory address buffer (bit 6 is stored in the MS bit of the memory address);

- if the value is one, the unit will act in the following way:

  the data bits b6 - b0 are interpreted as the 7 LS bits of the memory address and are stored in the LS part of the memory address buffer;

  it will put the data (8 bits) stored in the memory on the address specified by the memory address buffer, into the data buffer and the data buffer empty bit in the slave status byte is set to '1' (the data buffer is the buffer that can be accessed by "read data" defined by the control bits);

  if a memory address is accessed that is not specified, the value "all zeros" will be put into the data buffer. The value "all zeros" will also be put into the memory address buffer.

4.0 DATA

When the control bits specify "read data", the data available in the slave's data buffer is received by the master.

The interpretation of the received data depends upon the specification of the unit and/or the contents of the memory (see this supplement, section 3.0).

When the control bits specify "write data", the interpretation of the received data depends on specification of this unit.

5.0 COMMANDS

When the control bits specify "write command", the following codes indicate the type of function the slave is required to perform. The "locking" or "unlocking" of a slave to a master is also specified in the control bits.

5.1 Command codes applicable to the address space 256 - 511

5.1.1 Command codes applicable to any type of equipment

Code	General commands	Audio commands	Video commands
0-9	Numerical values 0-9		
12	Unit standby		
13		Mute/demute (toggle)	
14	All analogue to nominal values		
15	Display		
16		Sound+	
17		Sound-	
18			Brightness+
19			Brightness-
20			Colour saturation+
21			Colour saturation-
22		Bass+	
23		Bass-	
24		Treble+	
25		Treble-	
26		Balance right+	
27		Balance left+	
49	Correction, clear		
53	Play		
54	Stop		
55	Record		
56	Connect (audio + video)		
57	Disconnect (audio + video)		
61	System standby		
64			Contrast+
65			Contrast-
66		Medium+	
67		Medium-	
68			Hue+
69			Hue-
70-79	not yet allocated (reserved for continuous functions)		
127	Unit mains off		
128	Block no. 0		
"	" "		
159	Block no. 31		
160	Unit no. 0 within a block		
"	" " " "		
"	" " " "		
167	Unit no. 7 within a block		

Code	General commands	Audio commands	Video commands
192	Key release		
193	Command extension		
194			
195			
196		Mute	
197		Demute	
198	Set protection		
199	Reset protection		
200		Stereo sound	
201		Mono sound	
202		Two independent sound channels	
203		Select channel A	
204		Select channel B	
205	Set external (audio + video)		
206	Set external only audio		
207	Set external only video		
208	Set internal (audio + video)		
209	Set internal only audio		
210	Set internal only video		

5.1.2 Command codes applicable to specified types of equipment

5.1.2.1 Recorders and players

Code	Command
30	Search forward to next marker
31	Search reverse to next marker
34	Play slow reverse
37	Step picture reverse
38	Slow motion speed+
39	Slow motion speed-
40	Play slow forward
41	Step picture forward
42	Play fast forward
44	Picture search reverse
45	Eject
46	Picture search forward
47	Play normal reverse
48	Pause
50	Tape rewind
52	Tape wind
59	Arm up
60	Arm down

5.1.2.2 Tuners (Video & Audio)

Code	Command
30	Search forward
31	Search reverse

5.1.2.3 Source selector

Code	Command
211	Connect audio
212	Connect video
213	Disconnect audio
214	Disconnect video

5.2 Reserved codes for future standardization

All command codes >215 and ⩽255 are reserved for future standardization

5.3 Command procedures

Commands can be transmitted to a unit via the D²B in two ways:

1. Only the code belonging to the command is transmitted.
2. A sequence of two commands is transmitted as follows:

   - the code representing the command
   - the command "key-release" (code 192).

   The following command codes should always be followed by a key-release command:

   - the codes applicable to any type of equipment:

     codes 16 - 27
     codes 64 - 79

   - and the video recorder, video player command codes:

     code 38
     code 39.

If a certain command code has to be interpreted with another non-standardized meaning, such a command has to be preceded by the command code 193 (command extension).

Archief
Ir. A.P.M.Moelands

5.3 Command procedures

Commands can be transmitted to a unit via the D2B in two ways:

1. Only the code belonging to the command is transmitted.
2. A sequence of two commands is transmitted as follows:

   - the code representing the command
   - the command "key-release" (code 192).

The following command codes should always be followed by a key-release command:

- the codes applicable to any type of equipment:

  codes 16 - 27
  codes 64 - 79

- and the video recorder, video player command codes:

  code 38
  code 39.

If a certain command code has to be interpreted with another non-standardized meaning, such a command has to be preceded by the command code 193 (command extension).

For a source selector, it is necessary to define a procedure for making/breaking a connection between two units. This procedure is defined by the following sequence of commands :

- command indicating a block number (128-159)
- command indicating a unit number within a block (160-167)
- connect/disconnect command (56, 211 or 212)
- command indicating a block number (128-159)
- command indicating a unit number within a block (160-167).

A block is a part of the address space set aside for a particular type of equipment (see Supplement 2).

The sequence of the first two commands uniquely defines a unit designated as the "source unit".

The sequence of the last two commands uniquely defines a unit designated as the "destination unit".

Supplement 2

Address allocation:

1.0 ADDRESS SPACE DEFINITION

Different units are defined by the 12 address number bits providing a sequence of codes numbered 0-4095 with the following interpretation :

0 - 255	not defined
256 - 511	reserved for peripheral equipment and audio-video source selectors
512 - 4095	not defined.

2.0 ALLOCATION OF ADDRESS CODES

The address space 256-511 is divided into 32 blocks, each block indicating the type of equipment. Each block can define 8 units.

If only one unit in a certain block is present in the system, it will have the lowest address within the block. If a second unit in the same block is present in the system, it will have the address "lowest + 1" within the block, and so on.

Bit 11 is the MSB of the address and bit 0, the LSB. The following allocation is defined :

- The most significant 4 address bits have the fixed value:

bit 11 = 0
bit 10 = 0
bit 9 = 0
bit 8 = 1

- The block numbers indicated by bits 7, 6, 5, 4 and 3 are allocated to the following types of equipment:

bit no. 7	6	5	4	3	allocated type of equipment
0	0	0	0	0	video monitor
0	0	0	0	1	video pre-amp
0	0	0	1	0	video source selector
0	0	0	1	1	audio source selector
0	0	1	0	0	video tuner
0	0	1	0	1	video recorder
0	0	1	1	0	video player
0	0	1	1	1	video camera
0	1	0	0	0	teletext decoder
0	1	0	0	1	videotex decoder
0	1	0	1	0	reserved for future standardization
0	1	0	1	1	reserved for future standardization
0	1	1	0	0	reserved for future standardization
0	1	1	0	1	reserved for future standardization
0	1	1	1	0	reserved for future standardization
0	1	1	1	1	free for use selector
1	0	0	0	0	audio tuner
1	0	0	0	1	audio recorder
1	0	0	1	0	compact disc
1	0	0	1	1	phono
1	0	1	0	0	reserved for future standardization
1	0	1	0	1	reserved for future standardization
1	0	1	1	0	reserved for future standardization
"	"	"	"	"	"
1	1	1	1	0	reserved for future standardization
1	1	1	1	1	free for use

If a unit is defined which consists of several type of equipment then this unit will have a block number equal to the lowest block number of the equipment implemented within it. For example, a television set will have the block number 00000 because it contains a video monitor.

Archief
Ir. A.P.M.Moelands

©1984 Philips Export B.V

This information is furnished for guidance, and with no guarantee as to its accuracy or completeness; its publication conveys no licence under any patent or other right, nor does the publisher assume liability for any consequence of its use; specifications and availability of goods mentioned in it are subject to change without notice; it is not to be reproduced in any way, in whole or in part, without the written consent of the publisher.

Printed in The Netherlands

Date of release: February 1984

9398 616 60011