A

A-level
COMPUTER SCIENCE
Paper 2
7517/2

Insert

FIGURE 1 for use in answering Question 4

FIGURE 3 for use in answering Question 5

TABLE 1 for use in answering Question 6

FIGURE 4 for use in answering Question 6

[Turn over]

FIGURE 1

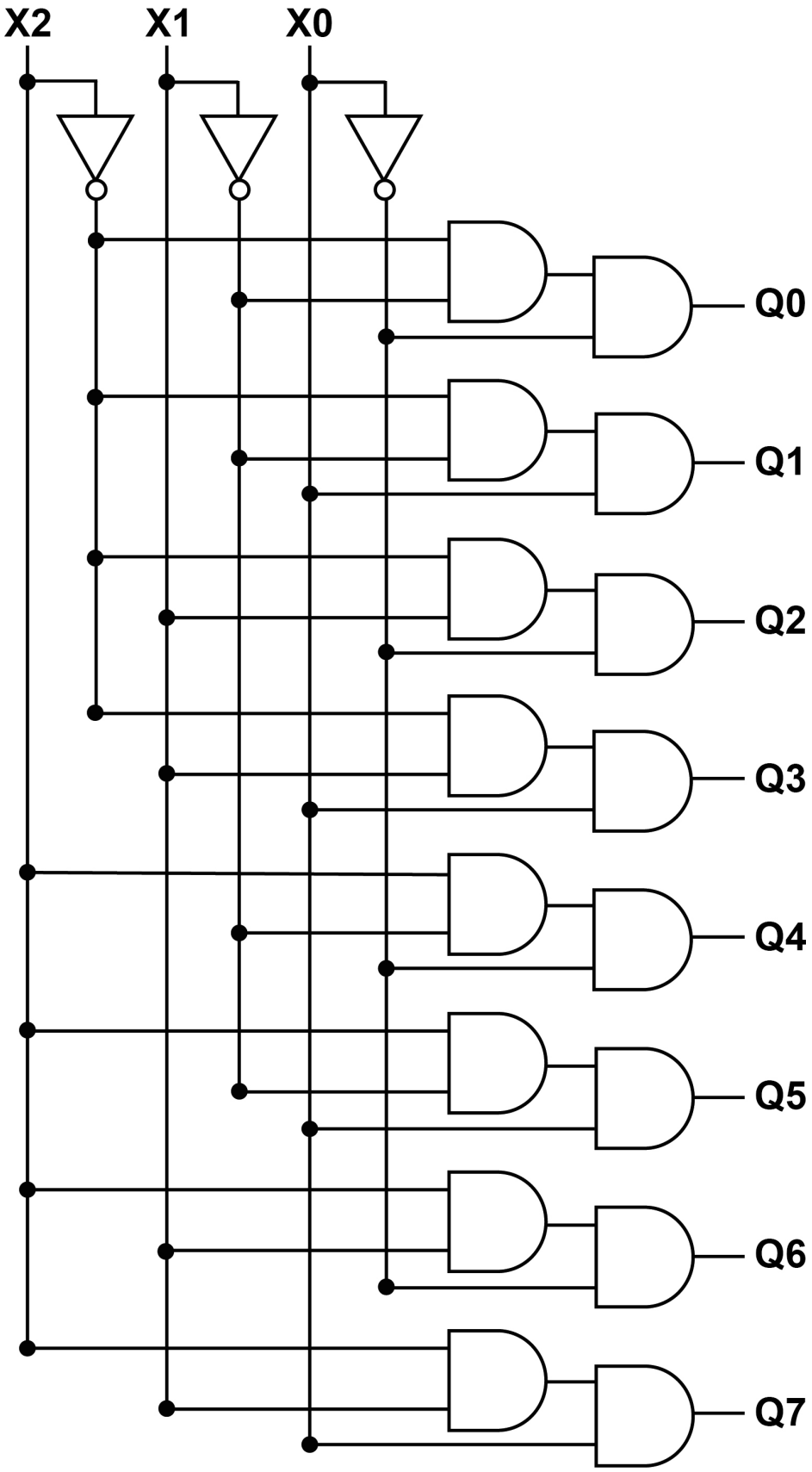

FIGURE 3

Facility(FacilityID, Description, MaxPeople, PricePerHour)
FacilityForSport(Sport, FacilityID)
Booking(FacilityID, BookingDate, StartTime, EndTime, CustomerID)
Customer(CustomerID, Forename, Surname, EmailAddress)

[Turn over]

TABLE 1

Standard AQA assembly language instruction set

LDR Rd, <memory ref>	**Load the value stored in the memory location specified by** <memory ref> **into register** d**.**
STR Rd, <memory ref>	**Store the value that is in register** d **into the memory location specified by** <memory ref>**.**
ADD Rd, Rn, <operand2>	**Add the value specified in** <operand2> **to the value in register** n **and store the result in register** d**.**
SUB Rd, Rn, <operand2>	**Subtract the value specified by** <operand2> **from the value in register** n **and store the result in register** d**.**
MOV Rd, <operand2>	**Copy the value specified by** <operand2> **into register** d**.**
CMP Rn, <operand2>	**Compare the value stored in register** n **with the value specified by** <operand2>**.**
B <label>	**Always branch to the instruction at position** <label> **in the program.**
B <condition> <label>	**Branch to the instruction at position** <label> **if the last comparison met the criterion specified by** <condition>**.** **Possible values for** <condition> **and their meanings are:** EQ**: equal to** NE**: not equal to** GT**: greater than** LT**: less than**

`AND Rd, Rn, <operand2>`	**Perform a bitwise logical AND operation between the value in register `n` and the value specified by `<operand2>` and store the result in register `d`.**
`ORR Rd, Rn, <operand2>`	**Perform a bitwise logical OR operation between the value in register `n` and the value specified by `<operand2>` and store the result in register `d`.**
`EOR Rd, Rn, <operand2>`	**Perform a bitwise logical XOR (exclusive or) operation between the value in register `n` and the value specified by `<operand2>` and store the result in register `d`.**
`MVN Rd, <operand2>`	**Perform a bitwise logical NOT operation on the value specified by `<operand2>` and store the result in register `d`.**
`LSL Rd, Rn, <operand2>`	**Logically shift left the value stored in register `n` by the number of bits specified by `<operand2>` and store the result in register `d`.**
`LSR Rd, Rn, <operand2>`	**Logically shift right the value stored in register `n` by the number of bits specified by `<operand2>` and store the result in register `d`.**
`HALT`	**Stops the execution of the program.**

BLANK PAGE

LABELS: A label is placed in the code by writing an identifier followed by a colon (:). To refer to a label, the identifier of the label is placed after the branch instruction.

Interpretation of `<operand2>`

`<operand2>` **can be interpreted in two different ways, depending on whether the first character is a # or an R:**

- `#` **– use the decimal value specified after the** `#`**, eg** `#25` **means use the decimal value 25**
- `Rm` **– use the value stored in register** `m`**, eg** `R6` **means use the value stored in register 6**

The available general-purpose registers that the programmer can use are numbered 0–12

[Turn over]

FIGURE 4

```
   CMP R2, #0
   BEQ exit
   MOV R0, #0
   MOV R3, #1
moveleft:
   LSL R2, R2, #1
   LSL R3, R3, #1
   CMP R2, R1
   BLT moveleft
   BEQ mainloop
   LSR R2, R2, #1
   LSR R3, R3, #1
mainloop:
   CMP R1, R2
   BLT skip
   ADD R0, R0, R3
   SUB R1, R1, R2
skip:
   AND R4, R3, #1
   CMP R4, #1
   BEQ skipshiftR2
   LSR R2, R2, #1
skipshiftR2:
   LSR R3, R3, #1
   CMP R3, #0
   BNE mainloop
exit:
   HALT
```

END OF SOURCE MATERIAL

BLANK PAGE

BLANK PAGE