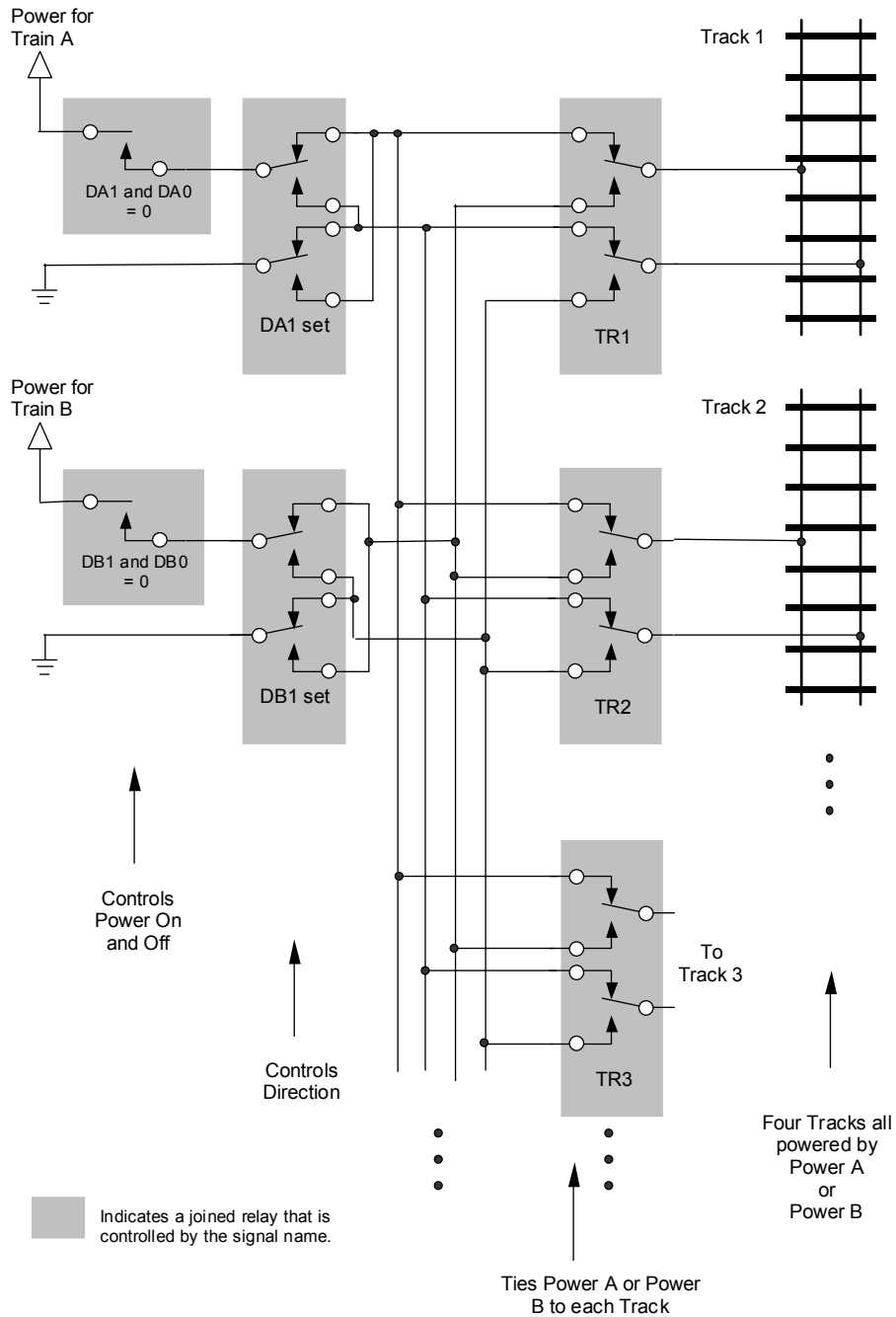


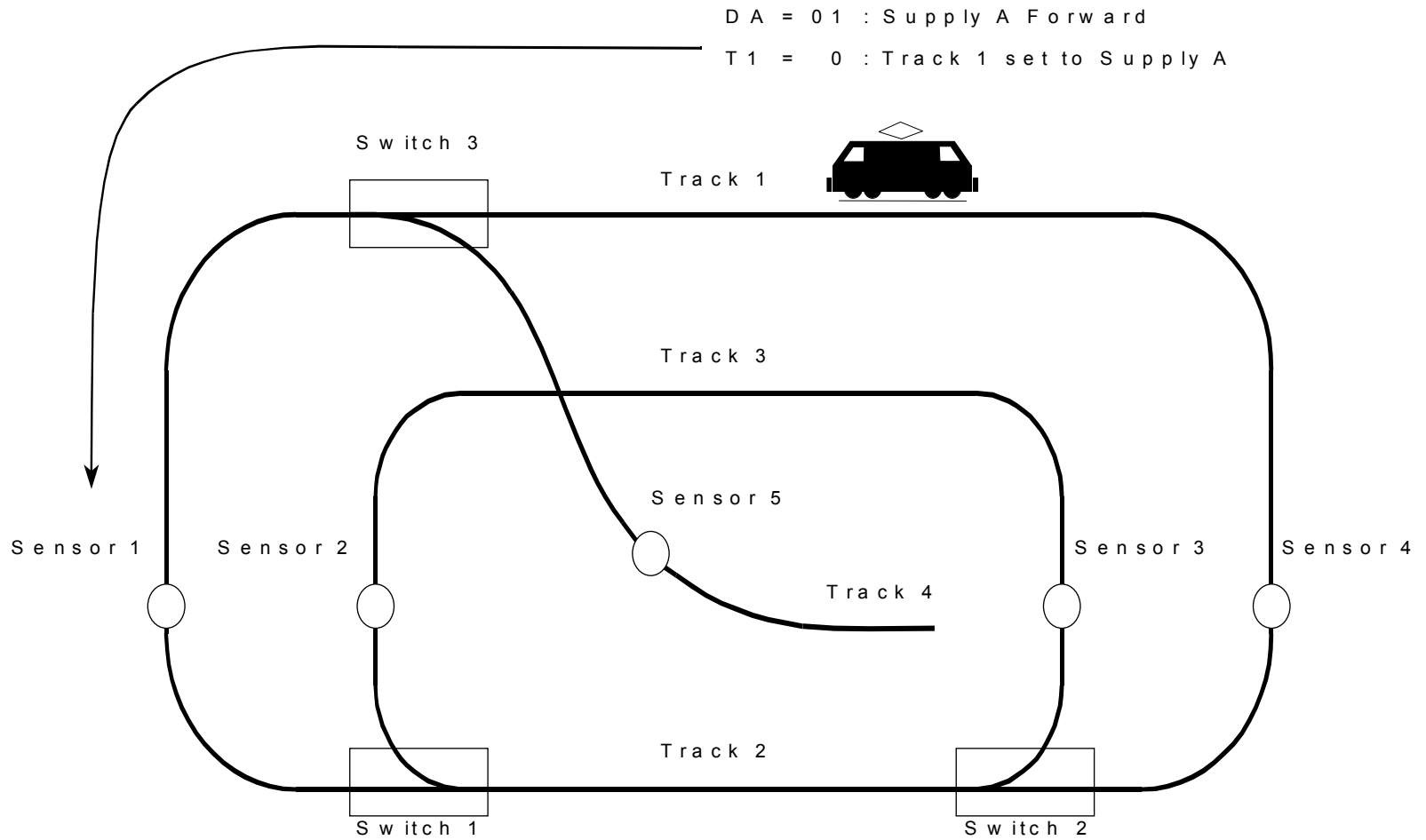
**Figure 8.1** Track Layout with Input Sensors and Output Switches and Output Tracks.



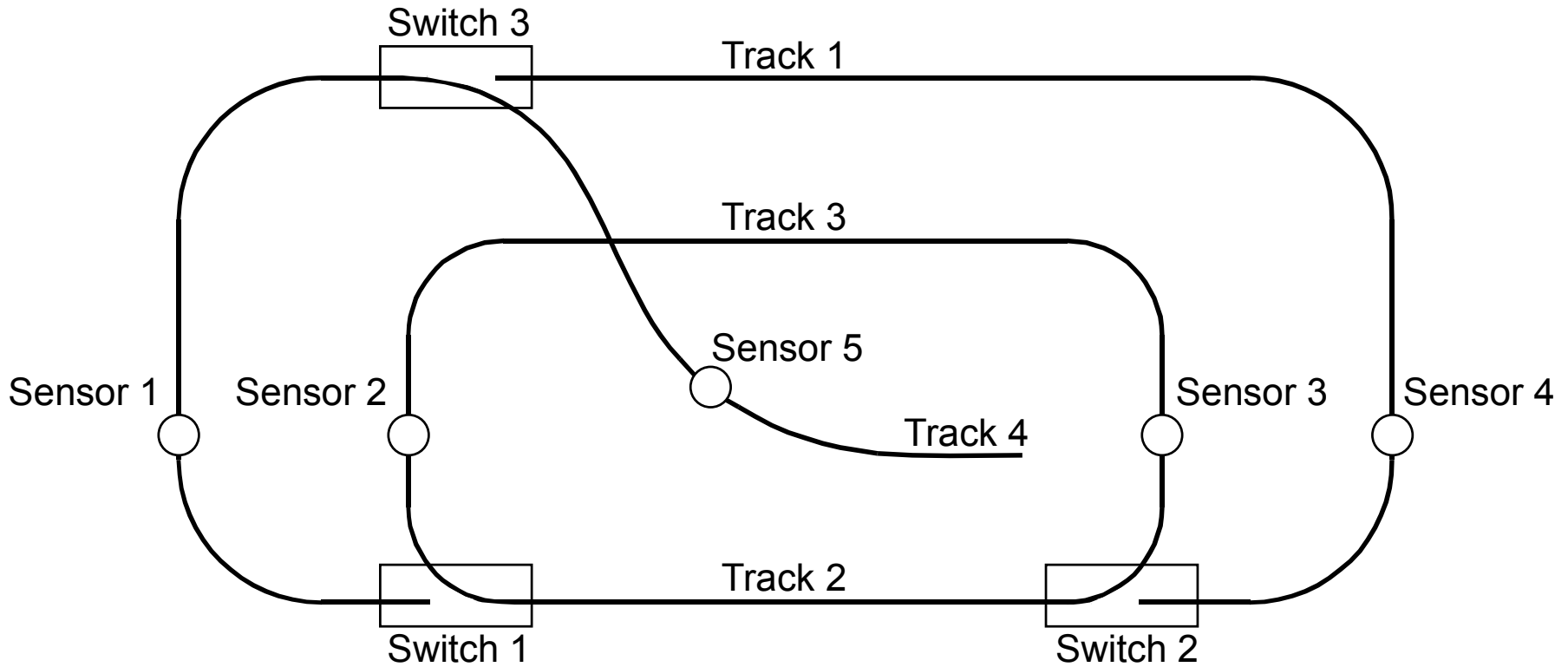
Indicates a joined relay that is controlled by the signal name.

Ties Power A or Power B to each Track

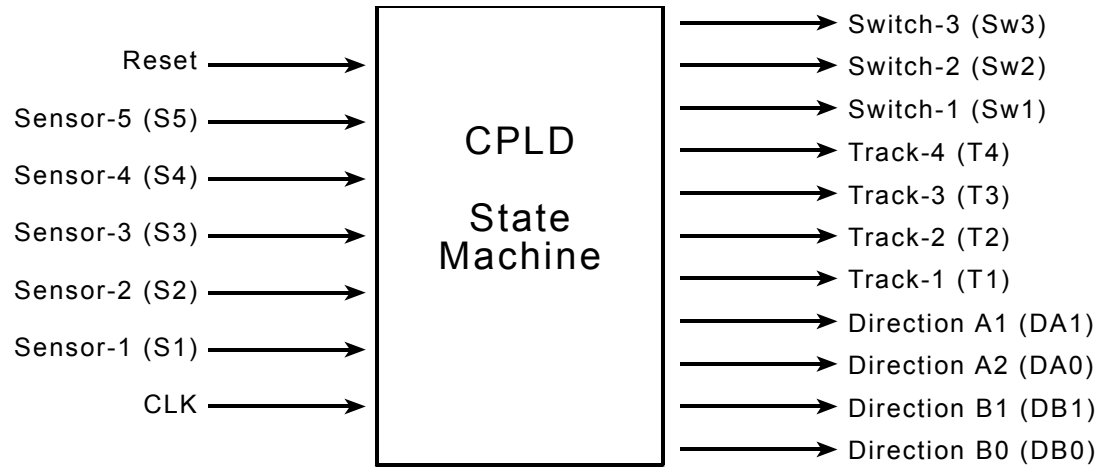
Four Tracks all powered by Power A or Power B



**Figure 8.3** Track Power is connected to one of Two Power Sources: A and B.



**Figure 8.4** Track Direction if all Switches are Asserted (SW1 = SW2 = SW3 = 1)



Sensor (S1, S2, S3, S4, S5) = 1 Train Present

= 0 Train not Present

Switches (SW1, SW2, SW3) = 0 Connected to Outside Track

= 1 Connected to Inside Track

Track (T1, T2, T3, T4) = 0 A Virtual Power on Track n

= 1 B Virtual Power on Track n

Direction (DA1-DA0) and (DB1-DB0) = 00 Stop

= 01 Forward (Counterclockwise)

= 10 Backward (Clockwise)

**Figure 8.5** State Machine I/O Configuration

## Description of States in Example State Machine

### All States

- T3 Asserted: The B power supply is assigned to track 3.
- All signals that are not "Asserted" are zero and imply a logical result as described.

### ABout: "Trains A and B Outside"

- DA0 Asserted: Train A is on the outside track and moving counter-clockwise (forward).
- DB0 Asserted: Train B is on the inner track (not the common track) and also moving forward.
- Note that by NOT Asserting DA1, it is automatically zero -- same for DB1. Hence, the outputs are DA = "01" and DB = "01".

### Ain: "Train A moves to Common Track"

- Sensor 1 has fired either first or at the same time as Sensor 2.
- Either Train A is trying to move towards the common track, or
- Both trains are attempting to move towards the common track.
- Both trains are allowed to enter here; however, state Bstop will stop B if both have entered.
- DA0 Asserted: Train A is on the outside track and moving counter-clockwise (forward).
- DB0 Asserted: Train B is on the inner track (not the common track) and also moving forward.

**Bstop:** "Train B stopped at S2 waiting for Train A to clear common track"

- DA0 Asserted: Train A is moving from the outside track to the common track.
- Train B has arrived at Sensor 2 and is stopped and waits until Sensor 4 fires.
- SW1 and SW2 are NOT Asserted to allow the outside track to connect to common track.
- Note that T2 is not asserted making Track 2 tied to the A Power Supply.

**Bin:** "Train B has reached Sensor 2 before Train A reaches Sensor 1"

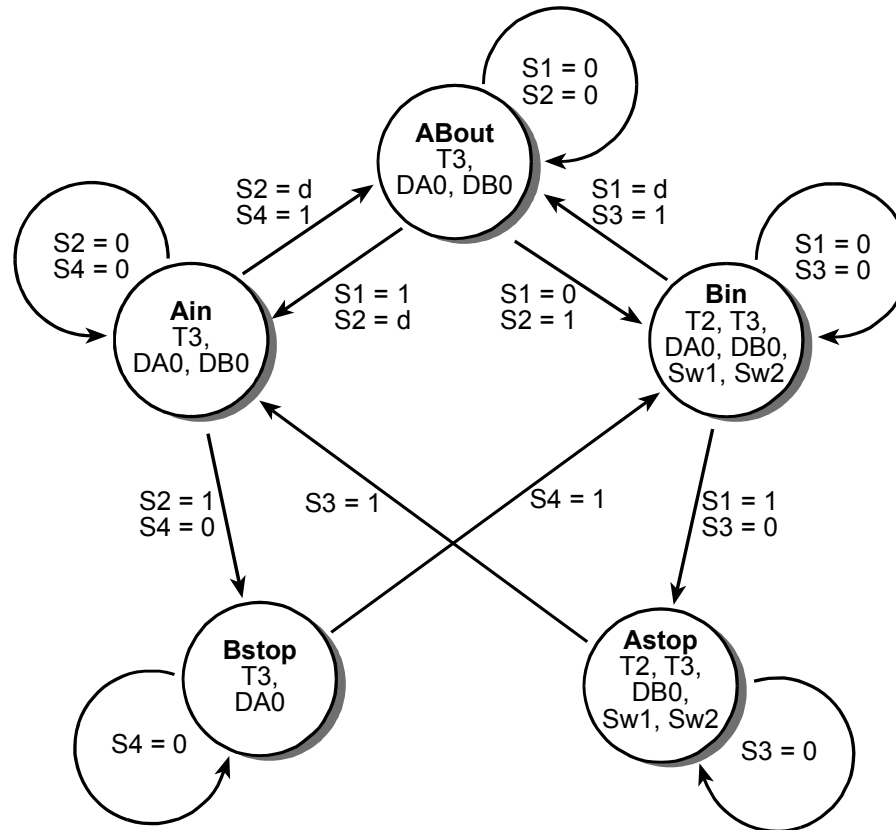
- Train B is allowed to enter the common track. Train A is approaching Sensor 1.
- DA0 Asserted: Train A is on the outside track and moving counterclockwise (forward).
- DB0 Asserted: Train B is on the inner track moving towards the common track.
- SW1 Asserted: Switch 1 is set to let the inner track connect to the common track.
- SW2 Asserted: Switch 2 is set to let the inner track connect to the common track.
- T2 Asserted: The B Power Supply is also assigned to the common track.

**Astop:** "Train A stopped at S1 waiting for Train B to clear the common track"

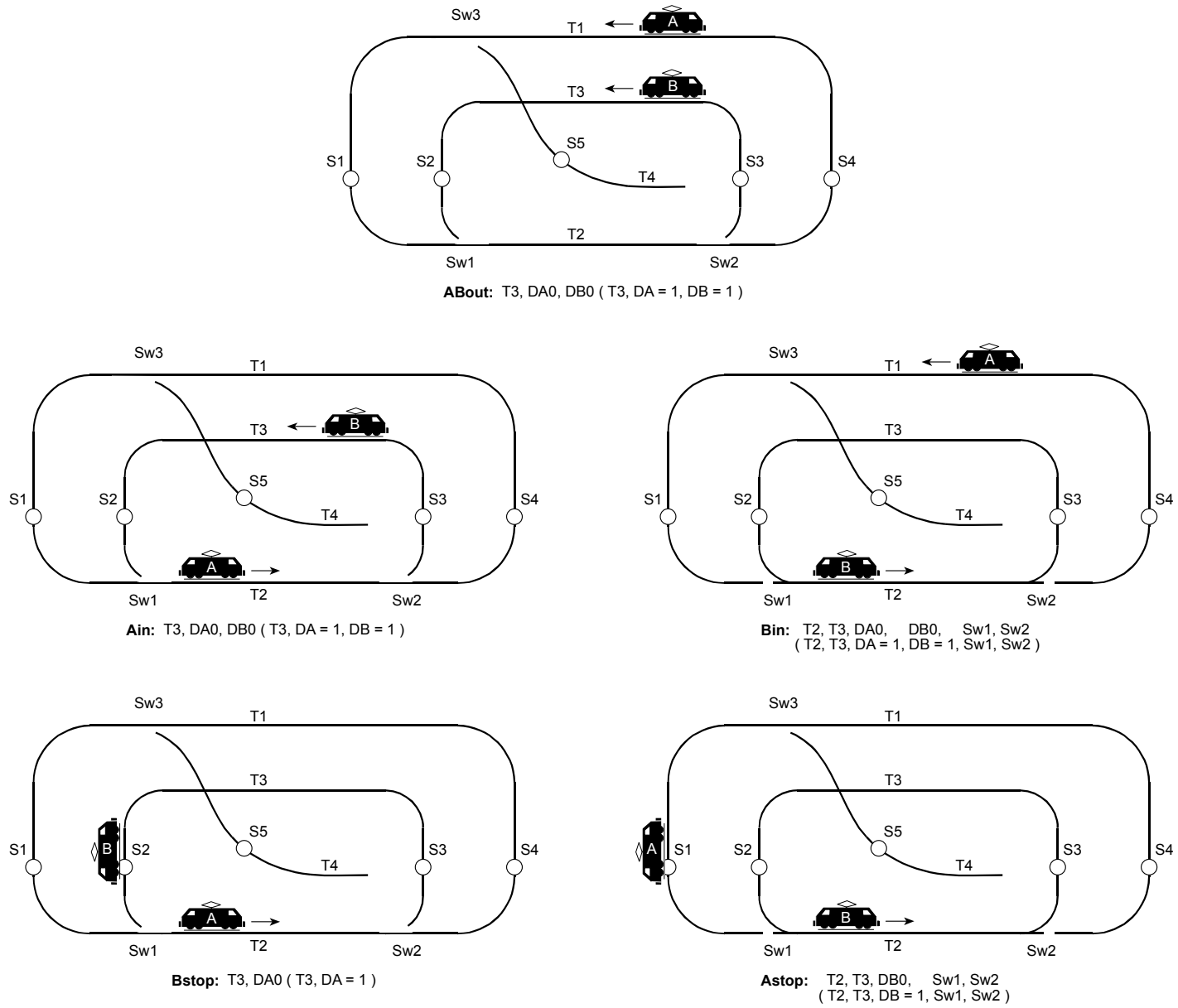
- DB0 Asserted: Train B is on the inner track moving towards the common track.
- SW1 and SW2 Asserted: Switches 1 and 2 are set to connect the inner track to the common track.
- T2 Asserted: The B Power Supply is also assigned to the common track.







**Figure 8.7** Example Train Controller State Diagram.



**Figure 8.8** Working diagrams of train positions for each state.

Table 8.1 Outputs corresponding to states.

<b>State</b>	<b>ABout</b>	<b>Ain</b>	<b>Astop</b>	<b>Bin</b>	<b>Bstop</b>
Sw1	0	0	1	1	0
Sw2	0	0	1	1	0
Sw3	0	0	0	0	0
T1	0	0	0	0	0
T2	0	0	1	1	0
T3	1	1	1	1	1
T4	0	0	0	0	0
DA(1-0)	01	01	00	01	01
DB(1-0)	01	01	01	01	00

```

-- Example State machine to control trains-- File: Tcontrol.vhd
-- These libraries are required in all VHDL source files
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
    -- This section defines state machine inputs and outputs
    -- No modifications should be needed in this section
ENTITY Tcontrol IS
PORT(
    reset, clock, sensor1, sensor2,
    sensor3, sensor4, sensor5 : IN   STD_LOGIC;
    switch1, switch2, switch3 : OUT STD_LOGIC;
    track1, track2, track3, track4   : OUT   STD_LOGIC;
    -- dirA and dirB are 2-bit logic vectors(i.e. an array of 2 bits)
    dirA, dirB                       : OUT STD_LOGIC_VECTOR( 1 DOWNTO 0 ));
END Tcontrol;
    -- This code describes how the state machine operates
    -- This section will need changes for a different state machine
ARCHITECTURE a OF Tcontrol IS
    -- Define local signals (i.e. non input or output signals) here
    TYPE STATE_TYPE IS ( ABout, Ain, Bin, Astop, Bstop );
    SIGNAL state: STATE_TYPE;
    SIGNAL sensor12, sensor13, sensor24 : STD_LOGIC_VECTOR(1 DOWNTO 0);
BEGIN
    -- This section describes how the state machine behaves
    -- this process runs once every time reset or the clock changes
    PROCESS ( reset, clock )
    BEGIN
        -- Reset to this state (i.e. asynchronous reset)
        IF reset = '1' THEN
            state <= ABout;
        ELSIF clock'EVENT AND clock = '1' THEN

```

```

-- clock'EVENT means value of clock just changed
--This section will execute once on each positive clock edge
--Signal assignments in this section will generate D flip-flops
-- Case statement to determine next state
CASE state IS
  WHEN ABout =>
    -- This Case checks both sensor1 and sensor2 bits
    CASE Sensor12 IS
      -- Note: VHDL's use of double quote for bit vector versus
      -- a single quote for only one bit!
      WHEN "00" => state <= About;
      WHEN "01" => state <= Bin;
      WHEN "10" => state <= Ain;
      WHEN "11" => state <= Ain;
      -- Default case is always required
      WHEN OTHERS => state <= ABout;
    END CASE;
  WHEN Ain =>
    CASE Sensor24 IS
      WHEN "00" => state <= Ain;
      WHEN "01" => state <= ABout;
      WHEN "10" => state <= Bstop;
      WHEN "11" => state <= ABout;
      WHEN OTHERS => state <= ABout;
    END CASE;
  WHEN Bin =>
    CASE Sensor13 IS
      WHEN "00" => state <= Bin;
      WHEN "01" => state <= ABout;
      WHEN "10" => state <= Astop;
      WHEN "11" => state <= About;
      WHEN OTHERS => state <= ABout;
    END CASE;
  WHEN Astop =>
    IF Sensor3 = '1' THEN
      state <= Ain;
    ELSE
      state <= Astop;
    END IF;
  WHEN Bstop =
    IF Sensor4 = '1' THEN
      state <= Bin;
    ELSE
      state <= Bstop;
    END IF;
END CASE;
END IF;
END PROCESS;

```

```

-- combine sensor bits for case statements above
-- "&" operator combines bits
sensor12 <= sensor1 & sensor2;
sensor13 <= sensor1 & sensor3;
sensor24 <= sensor2 & sensor4;

-- These outputs do not depend on the state
Track1 <= '0';
Track4 <= '0';
Switch3 <= '0';

-- Outputs that depend on state, use state to select value
-- Be sure to specify every output for every state
-- values will not default to zero!

WITH state SELECT
    Track3 <=
        '1'    WHEN ABout,
        '1'    WHEN Ain,
        '1'    WHEN Bin,
        '1'    WHEN Astop,
        '1'    WHEN Bstop;

WITH state SELECT
    Track2 <=
        '0'    WHEN ABout,
        '0'    WHEN Ain,
        '1'    WHEN Bin,
        '1'    WHEN Astop,
        '0'    WHEN Bstop;

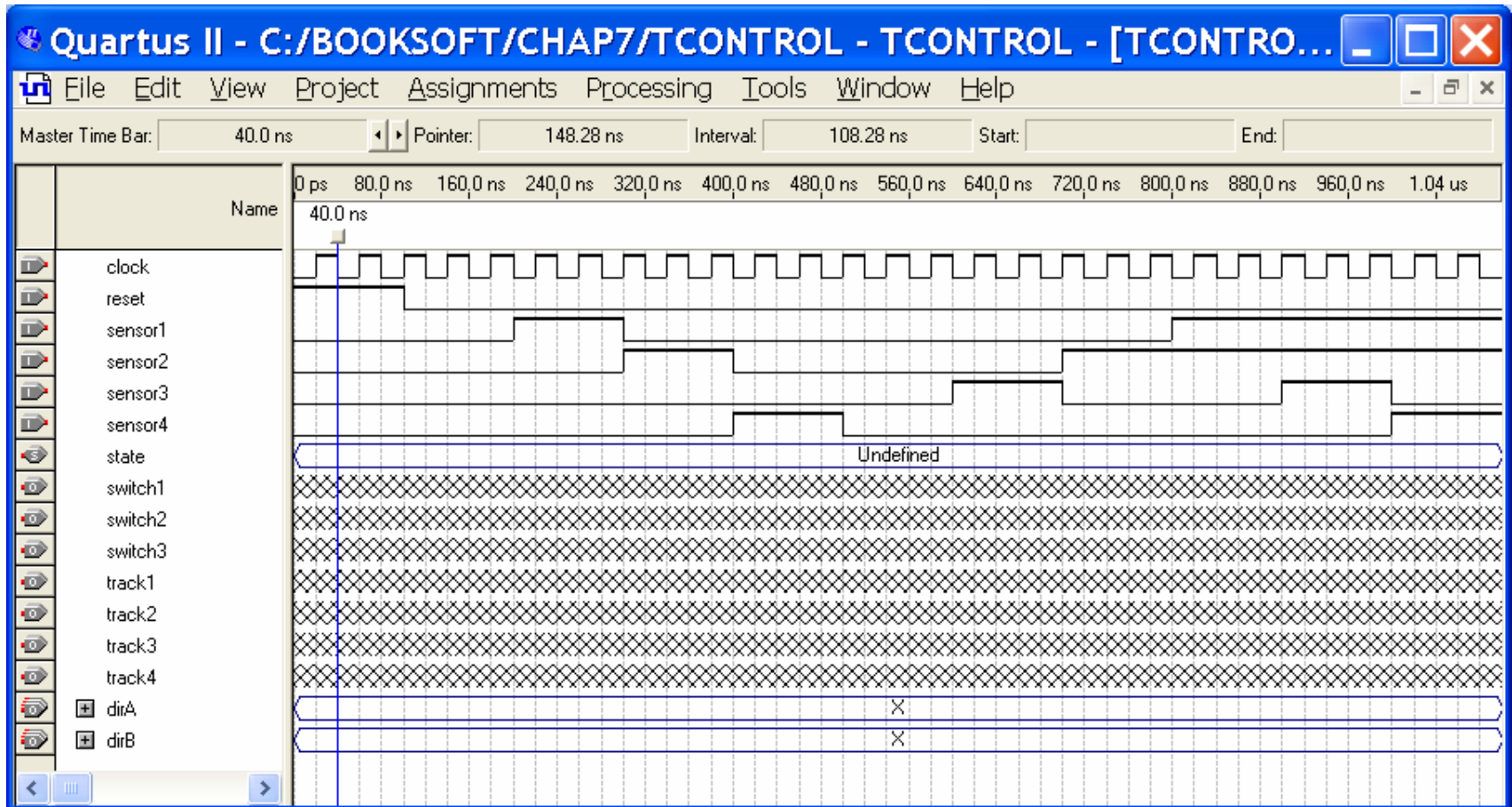
WITH state SELECT
    Switch1 <=
        '0'    WHEN ABout,
        '0'    WHEN Ain,
        '1'    WHEN Bin,
        '1'    WHEN Astop,
        '0'    WHEN Bstop;

WITH state SELECT
    Switch2 <=
        '0'    WHEN ABout,
        '0'    WHEN Ain,
        '1'    WHEN Bin,
        '1'    WHEN Astop,
        '0'    WHEN Bstop;

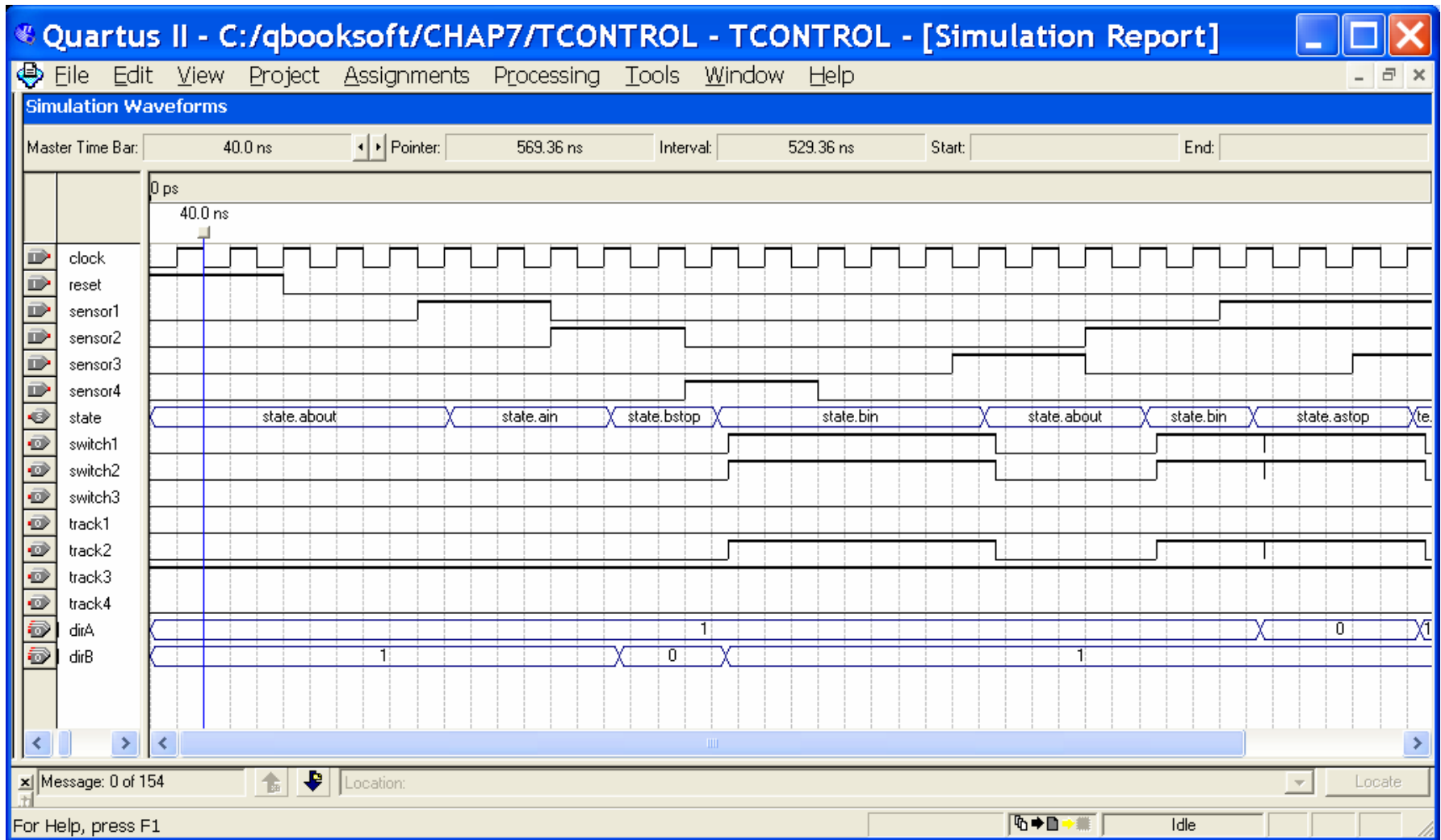
WITH state SELECT
    DirA <=
        "01"   WHEN ABout,
        "01"   WHEN Ain,
        "01"   WHEN Bin,
        "00"   WHEN Astop,
        "01"   WHEN Bstop;

WITH state SELECT
    DirB <=
        "01"   WHEN ABout,
        "01"   WHEN Ain,
        "01"   WHEN Bin,
        "01"   WHEN Astop,
        "00"   WHEN Bstop;

END a;
```

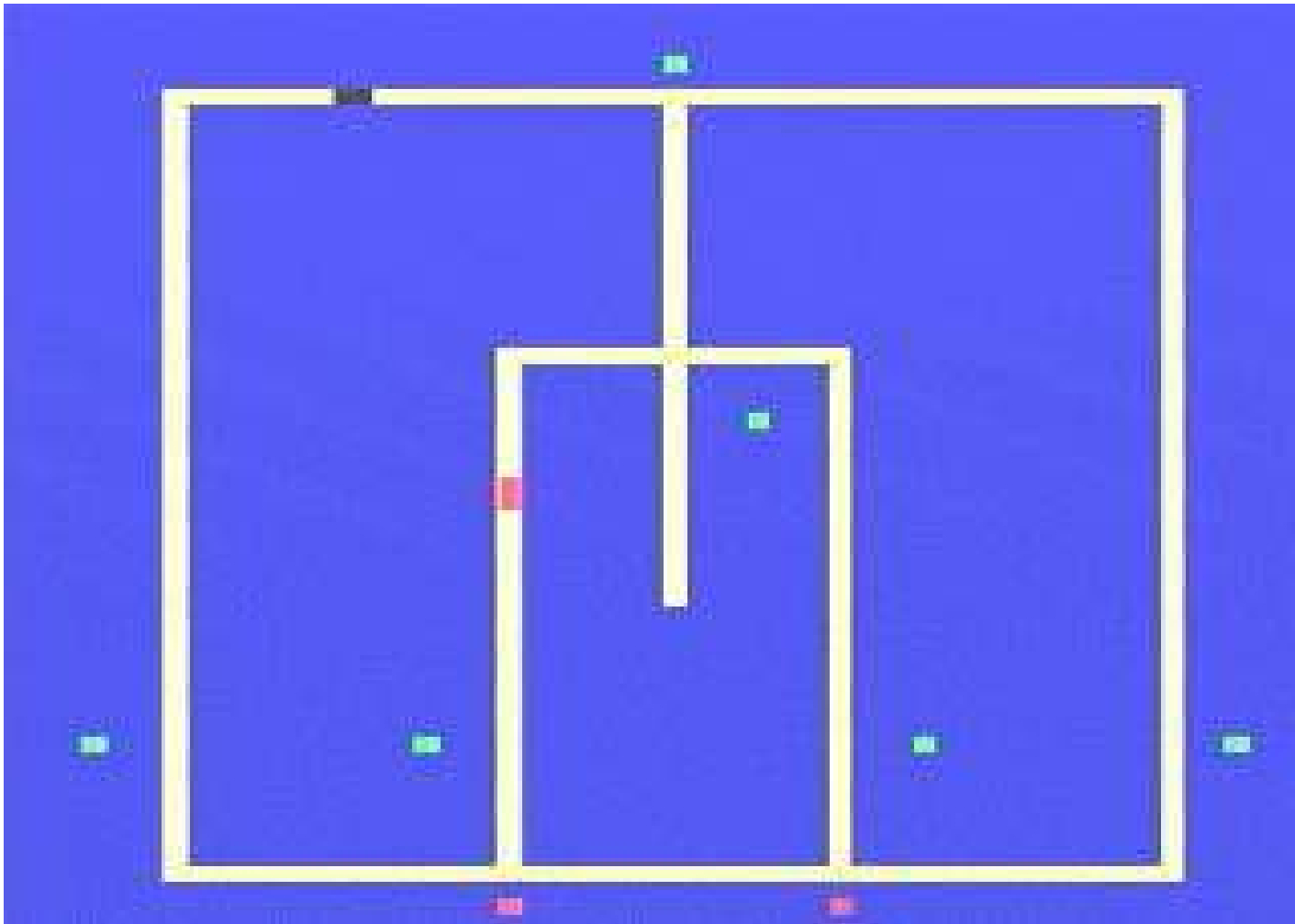


**Figure 8.9** Tcontrol.vwf vector waveform file for simulation.



**Figure 8.10** Simulation of Tcontrol.vhd using Tcontrol.vec vector file.





**Figure 8.11** Video Image from Train System Simulation.