IN4387 System Validation

Design & Verification of Controller for a Package Storage System

S. Balasubramanian, #0785610
Voudouris. P, #0788565
Gozek. E, #0786244

Eindhoven University of Technology
Department of Embedded Systems

September 23, 2012

Contents

Chapter 1

Introduction

The project described in this report discusses the design and verification of a small packet storage system. The packet storage system is inspired by the distributed controller of an operational product manufactured by Vanderlande, a Dutch company. The aim of this project is to understand and formulate the requirements of such a system, design the system and in the end verify that the system would meet the proposed requirements when in operation under any circumstance.

The packet storage system consists of two conveyor belts, one for receiving a packet into the system and the other for delivering the packet out of the system. The system consists of several racks where packets can be stored. The packets are stored by means of two elevator platforms (situated serially, one above the other). Moreover, there are five controllers in the system which run in parallel. Figure 1.1 shows the diagram of such a system. The controller C1 controls the input conveyor belt, controller C2 controls the output conveyor belt, conveyor C3 and C4 operate its respective elevators and controller C5 keeps track of all the information about racks, packet storage and its dispatch.

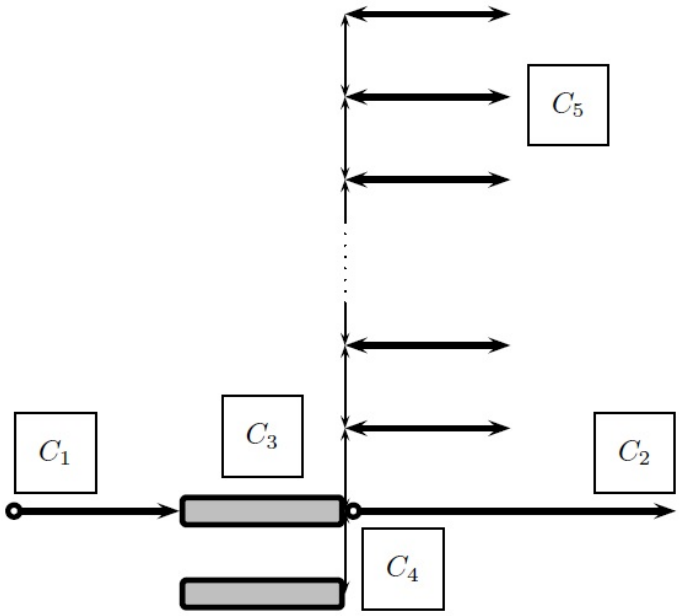

Figure 1.1: Packet storage system, courtesy [1]

There are however, certain constraints to be understood for the given system. For example, the number of packets on the conveyor belt, rack slots and elevator platform can carry only one packet at a time, etc. Further constraints are discussed in the next section under global requirements.

The sections described in the report are the global requirements, external interactions in the system, translated requirements, the architecture of the system transpired, discussion on controllers with its labelled transition diagram(LTS) and finally discuss the verification of its global requirements. The system is to be modelled to be intelligent to fulfil the basic requirements desired from such a system.

Chapter 2

Global requirements

Global Requirements

In the section, we describe the global requirements to required initially for the design of the controller:

1. Each elevator, rack and conveyor belt contains at most one packet.
2. Packet is exchanged only when the elevator platform is at the same level as that of a conveyor belt.
3. Packet is exchanged only when elevator platform is at the same level as that of a rack.
4. The two elevators cannot be at the same position.
5. The lower elevator must never pass the upper one.
6. Packets are always delivered in the same order as requested.
7. If a packet is ready to enter and there is a free position at the rack(s), it will be eventually accepted.
8. If a requested packet is in the system, it will be eventually delivered.
9. If a packet is unable to be located, a unique alarm must be generated.
10. The number of packets in the system can at most be equal to the number of racks.

Chapter 3

External interactions

This section lists the external interactions for the packet storage system. These are the high actions that are essential for the design of the system. The table 3.1 lists the interactions in the system, with first column denoting the name of the action, second column lists its description and the third column lists the parameters essential for that action. The internal actions within the system are described in a later section.

Note: The parameters for the actions can be shared, which means that they are part of two different actions which need to be synchronized.

Action	Description	Parameter
WantIn	Environment requires a packet to be inserted	packetID
QueryRackSpace	Queries if a position is available in racks	bool
InputConveyor_status	Queries if the input conveyor has an empty space	state
GetPacket	Gets a packet and places it onto conveyor belt	packetID
ElevatorStatus	Indicates the state and location of elevator	elevID, state
ConveyortoElevator_Comm	Load a packet from input conveyor to elevator	packetID, elevID
ElevatortoRack_Comm	Load a packet from elevator to rack	elevID, rackID, packetID
RacktoElevator_Comm	Load a packet from rack to elevator	elevID, rackID, packetID
ElevatortoConveyor_Comm	Load a packet from elevator to output conveyor	elevID,
OutputConveyor_status	Queries if the output conveyor has an empty space	state
WantOut	Environment requires a packet from the system	packetID
RequestPacket	Receive request for a packet	packetID
DeliverPacket	Delivers a packet to the system	packetID
RackStatus	Get number of packets currently in the system	Nat
QueryPacket	(Elevator) queries the position of a packet on rack	packetID
GetElevatorPos	Queries the position of an elevator	elevID, pos
MoveElevator	Moves the elevator to a rack position	elevID, rackID

Table 3.1: External interactions: Packet storage system

Chapter 4

Translated requirements

This section lists the requirements from section 2 in terms of interactions described in section 3.

1. *Each elevator, rack and conveyor belt contains at most one packet*
    - There is a packet on the conveyor belt, so no packet is allowed to enter.
    - There is no packet on the conveyor belt, so no packet is allowed to leave.
    - There is a packet on the elevator, so no packet is allowed to be loaded.
    - There is no packet on the elevator, so no packet is allowed to be unloaded.
    - There is a packet on the rack, so no packet is allowed to be stored.
    - There is no packet on the rack, so no packet is allowed to leave.
2. *Packet is exchanged only when the elevator platform is at the same level as that of a conveyor belt*
    - If elevator platform is not on the same level of input conveyor belt, no packet is loaded onto the elevator.
    - If elevator platform is not the same level as the output conveyor belt, no packet is loaded onto the conveyor belt.
3. *Packet is exchanged only when elevator platform is at the same level as that of a rack*
    - If position of rack and elevator platform are different, no packet is stored onto the rack.
    - If position of rack and elevator platform are different, no packet is loaded onto the elevator platform from the rack.
4. *The two elevators cannot be at the same position*
    - It cannot happen that two elevators are at the same position.
5. *The lower elevator must never pass the upper one*
    - The lower elevator is always below the upper elevator.
    - The lower elevator cannot go to the highest position.
    - The upper elevator cannot go to the lowest position.

6. *Packets are always delivered in the same order as requested*
    - It is not possible to receive packet(s) in an order different than as requested.
7. *If a packet is ready to enter and there is a free position at the rack(s), it will be eventually accepted*
    - If there is no free position in racks, packet must not be accepted.
    - If there is a free position in the racks and the packet is ready to enter, it must be eventually stored.
8. *If a requested packet is in the system, it will be eventually delivered*
    - If a packet is requested and it is in the system, it must be delivered eventually.
    - If a packet is requested and it is not in the system, no packet is delivered.
9. *If a packet is unable to be located, a unique alarm must be generated*
    - If the requested packet is not located in the racks, system is informed with an alarm.
10. *The number of packets in the system can at most be equal to the number of racks*
    - The number packets in the racks cannot be greater than the number of racks in the system.

Chapter 5

Architecture

Chapter 6

Modelling the controller

Chapter 7

Verification

Chapter 8

Experimental results

Chapter 9

Conclusions and recommendations

Bibliography

[1] Project statement.

Appendix A

Source Code Structure