Contents● Hardware-software-partitioning  
● Identification/partitioning into individual tasks  
● Timeline for implementation and testing & Distribution of the individual  
tasks among group members

**Hardware-software-partitioning**Hardware:  
● Displaying characters that are stored in a ring buffer  
● Issuing interrupts for different button functions  
● Issuing interrupts for CAN events (receiving messages)  
● Putting CAN frames on the CAN bus  
Software:  
● Implementing the display scrolling feature  
● Adding to and clearing the ring buffer  
● Updating the scrolling frequency  
● Encoding CAN frames

**Identification/partitioning into individual tasks**Hardware  
● Switch /Button controller  
● Component Testbench  
● Integrating CAN bus  
● Multiple CAN clients ( hardware instance)  
Software  
● Read /write - Switch /Button  
● Irq- button  
● Irq- CAN

**Problems  
Adding a new client:**• Assigning a unique ID for each client  
• sync entire buffer content to newly added devices  
**Client buffer coherency:**• Storing order in the scrolling display ring buffer is important. Race conditions for  
syncing clients might lead to misplaced data in the mentioned buffer.  
**Scrolling feature:**• Scrolling data that are less than 8 characters long.

**Problems  
Adding a new client:**• Using 1 button for triggering an interrupt handler that reads the status of 5 switches and set the  
client id to that value.  
• After acquiring an ID, the device asks the highest priority client for the buffer content, and that client  
will reply with the complete content of its buffer.  
**Client buffer coherency:**• delay writing to buffer until after CAN i/o is done sending the sync message over the CAN bus  
**Scrolling feature:**• the first character that is to be added by a client will be sent by the assembler along with 7 off  
characters

General Description  
➔ Design a distributed system where clients can synchronous the  
modified data through a CAN bus.  
➔ If any client changes any data those changes will be communicated  
to others client. Client can change data using switch and button.  
➔ If any change is made the interrupt controller will send irq signal to  
CPU to broadcast the changes over CAN bus.  
➔ LED patterns must be same for all the client connected to the CAN  
bus.

HW and SW  
***HW*** (most of the work is already done)  
● Switch and Button – We have to extend this for interrupt.  
● LED controller, LED buffer, LED timer, CAN bus, Interrupt controller.  
***SW:***● Configure top.vhd, config.vhd and peripherals.vhd for CAN.  
● Using assembly develop software to preform all the necessary  
operation such as adding data, clearing the data and setting the  
frequency. This software will also raise interrupt if any change is  
detected.

Communication  
Change in switch/Button  
Interrupt controller  
Core software modify data  
LED buffer  
LED controller LED  
**Data change Transmit Over CAN**CAN frame by Core  
Interrupt the CAN I/O  
Modified data to CAN buffer  
To CAN

Communication  
Frame of CAN bus  
To CAN I/O  
Interrupt  
Core  
LED buffer  
LED controller LE

VERIFICATION PLAN  
● Design a new client and verify their behavior using test bench and  
simulation.  
● Integrate a CAN controller into the design and validate whether it  
provides proper sync between the two clients.  
● Assure data set (5 or more different patterns) works as expected  
when it is modified,cleared and runs on a different frequency