CAN Nodes for Independent Joint Control of Manipulator

Shaurya Chandra (20BEE0314)

***Abstract*—The modular designed by robotic platform for pluggable. The circuit module connected via CAN bus can reduce the complexity of the robot control system design and increase the reliability and flexibility. Robot platform uses a distributed circuit control structure based on the 32-bit ARM Cortex-M3 development of STM32 series chips, which could meet the actual needs by adding modules, such as wireless communication module, sensor module, the joint motion control module. Meanwhile, a variety of forms of bionic robot could been established with the multi-purpose robot frame and PC software, for the aims of making all parts coordinated and realizing intelligent control of the complex actions.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Robots are distributed systems where different devices perform specific tasks and need to exchange data to run the overall system. The distributed approach is becoming common also to encourage hardware reuse, thus speeding up the development of new robotic systems, as it has been done since years by software developers. In a distributed architecture, a key aspect is how communication requests are handled: should they be triggered by time or by events? How priorities should be assigned to different data sources? Do we need a static or a dynamic network configuration? There are many fields, e.g., factory automation, automotive networking, or sensor networks, where answers to these questions can be easily picked out, and one approach to communication scheduling can be recognized as preferable. It is not so for robotic applications, where it is hard to define which communication paradigm is the best, as different requirements are needed by the different components of a complex robotic system. In this project, we build a robot platform of the circuit control system using CAN bus controller with a microcontroller STM32 series produced for the module's main controller, with each module through a CAN bus connection, multi-master multi-slave structure, solve the robot control complexity of the system design and production. This work also aims to build a modular plug-in architecture with different types of sensor modules

A. Time triggered and event triggered communication

The choice between time-triggered and eventtriggered communication paradigms to design realtime systems has been subject to a long debate (Kopetz, 1993; Obermaisser, 2004), which highlighted advantages and drawbacks of both approaches. A time-triggered design leads to predictable temporal behavior, since each communication occurrence is planned at design time. This requires a detailed analysis of the overall system, which adds complexity to its design and limits further system expansions. On the other side, a-priori scheduling leads to temporal determinism of communication, guaranteeing quality of service. To improve flexibility, a centralized scheduler can be exploited performing online admissibility control. Depending on the scheduling policy adopted, an online scheduler can lead to high computational requirements. On the other hand, an event-triggered design does not need a-priori knowledge about the system to schedule communication, thus leading to a much more flexible design. As a consequence of such a flexibility, a much more extensive testing is required to verify that the system can handle communication requests under different load situations. An advantage of event-triggered designs is, generally, a better resource exploitation with respect to time-triggered designs, with higher achievable throughputs.

1) Communication Requirements in Robotic Systems: A first source of communication in a distributed robotic system are control loops, which need to exchange data between sensors and actuators. These data transfers are, generally, periodic, deterministic and known at design time. Control loops are highly affected by the presence of jitter (Perez et al., 2003), ‘ which introduces variable delay and, thus, may induce overshoots of the control action and instability. Another common source of data transmission is the broadcast of system status, like a heartbeat signal, to suddenly halt the system if a failure happens. The system status could be updated asynchronously on change, but updating it periodically is generally a safer solution: if the status is not received, every component of the system can enter a safe mode. This kind of periodic communication are best handled in a timetriggered way, to schedule transmission occurrences, preventing collisions and reducing jitter.

Besides the sensors used in control loops, in a robotic system, we find other kinds of data sources; for instance, proximity sensors and bumpers produce useful data only when triggered by some event, like approaching an obstacle, and the most important factor is to react to new reading as quick as possible. If sensor readings are transmitted periodically, the only way to reduce the worst case latency is to increase the update frequency, which leads to a waste of bandwidth when data are not relevant. Using eventtriggered transmissions saves bandwidth and, generally, reduces latency. Other sources of non periodic data are planners, which can be triggered by some event (e.g., a new goal) and their execution may be not constant in time. Again, to save bandwidth and reduce latency, an event-triggered messaging paradigm is preferable to transmit the output of a planner.

As a conclusion, in robotic systems both periodic, time-triggered, and sporadic, event-triggered, data transmissions are needed, thus a protocol which combines the two communication paradigms is desirable.

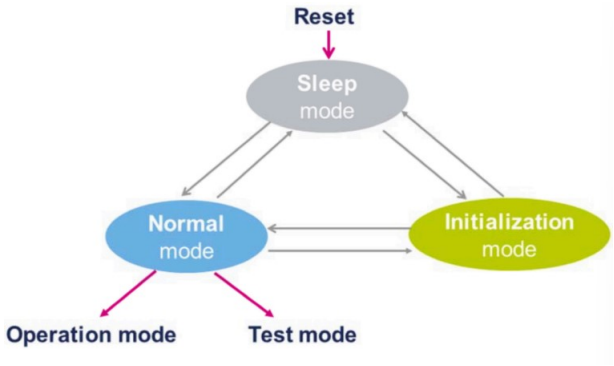
2) Scheduling Time- and EventTriggered Traffic on the CAN-bus: To connect the components of a distributed system a bus approach is preferable with respect to point-topoint architectures, reducing wiring complexity and making it easy to add devices to the network (Bonarini et al., 2011). Among the available bus communication standards, one of the most common is the CAN-Bus, which has been developed for automotive applications and is implemented by almost all microcontroller manufacturers, making it a good choice for distributed robotic systems. The CAN-Bus features a carrier-sense multipleaccess (CSMA) media access control (MAC), which is based on the ability of CAN controllers to detect the bus status while transmitting. Data are transmitted through a binary model of dominant and recessive bits; during the transmission of the arbitration field of a CAN frame, if the controller recognizes a dominant bit while it was trying to transmit a recessive one, it knows that the arbitration is lost and the node becomes a receiver. The CAN-Bus also focuses on fault detection and tolerance, providing hardware CRC calculation, automatic retransmission and many other features, as it was designed for automotive applications where reliability is a primary requirement. Due to hardware arbitration, the CAN-Bus is well suited for fixed-priority event-triggered communication, where high-priority messages win arbitration on low-priority ones; moreover, latency depends on bus load and a high-priority transmission can always be preempted by a higher-priority one. In order to extend CAN-Bus applications to distributed control systems, where temporal determinism and low jitter are mandatory, several protocols to schedule timetriggered traffic on the CAN-Bus have been presented and reviews are available

II. Implementation

STM32 Basic Extended Controller Area Network (bxCAN) is used for implementation. BxCAN is a standard serial differential bus broadcast interface, allowing the microcontroller to communicate with external devices connected to the same network bus using only two wires.

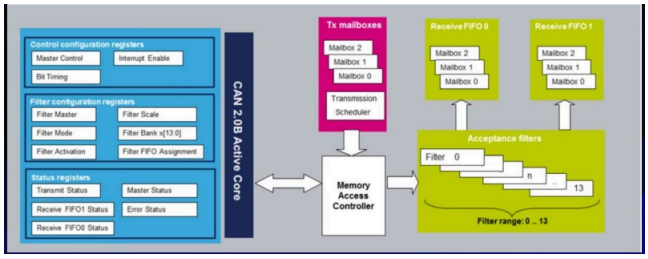
A. BxCAN Operating Modes

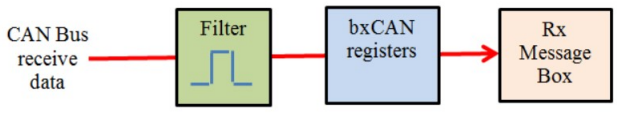
- Initialization, Normal and Sleep
- After a hardware reset, the BxCAN is in Sleep mode which operates at a lower power.
- The BxCAN enters Initialization mode via software to allow the configuration of the peripheral.
- Before entering Normal mode, the BxCAN must synchronize with the CAN bus, so it waits until the bus is idle (this means 11 consecutive recessive bits have been monitored on pin CANRX).
- When the CAN is in Normal mode, the user can select whether to run in Operation or Test mode.



B. BxCAN Block Diagram

- Three types of registers: Control configuration registers, filter configuration registers and status registers.
- Three transmit mailboxes are provided to the software for setting up messages. The Transmission Scheduler decides which mailbox has priority to be transmitted first.
- The BxCAN provides 14 scalable and configurable identifier filters for selecting the incoming messages the application needs and discarding the others.
- Two receive FIFOs: FIF00 and FIF01 are used by hardware to store incoming messages. Each FIFO can store three complete messages. The FIFOs are managed completely by hardware.





C. BxCAN Operation

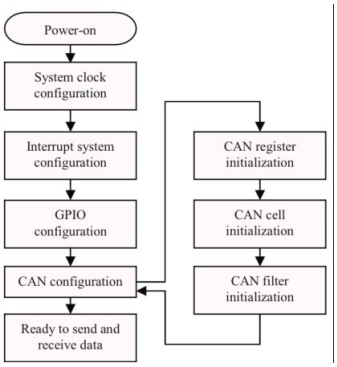
The following flow chart explains the BxCAN operation.

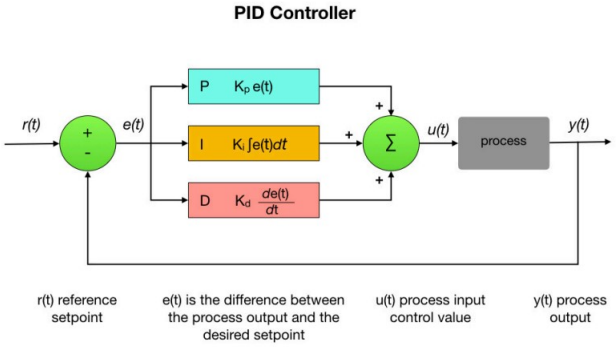
D. PID Motion Control

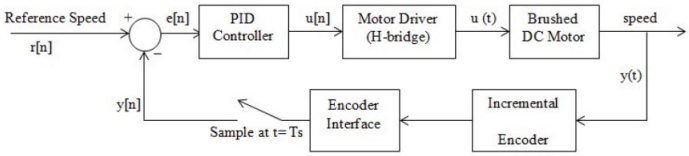
The following block diagrams explains the role of PID controller in motion control.

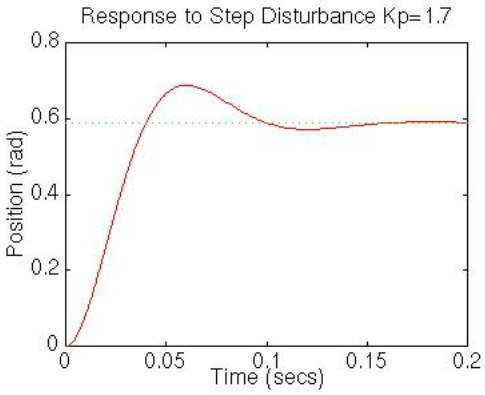
E. 3-D Mock-up of CAN Nodes

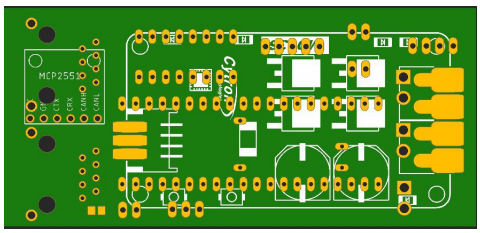
A 3-D mock up of CAN nodes is made using Autodesk Eagle and Autodesk Fusion 360.





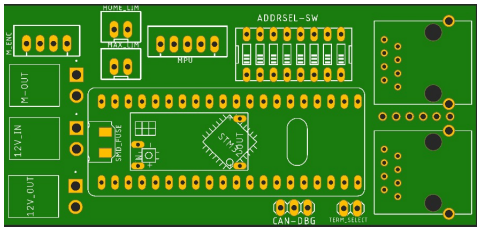


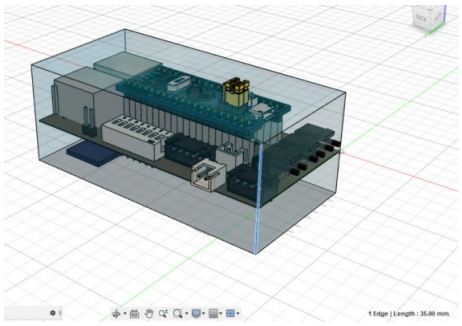




F. Hardware Requirements

- 12V Brushed DC Motor





- Motor Driver
- Microcontroller unit with CAN, Timers. (STM32F103 BluePill)
- CAN Transceiver

III. Conclusion

A. Advantages

- Easier maintenance
- Smaller number of fragile connections (Only 2 wires).
- Easier miniaturization of the system.
- Robust in electromagnetically noisy environments.

B. Challenges

- Hardware Complexity
- Difficult to implement

Acknowledgment

We would like to extend our warmest gratitude to our project guide Dr. Geethanjali P. for helping us and guiding us throughout the project.

References

[1] A Real-time CAN-bus Protocol for Robotic Applications.
[2] Ahmadzadeh, Hossein Masehian, Ellips Asadpour, Masoud. (2015). Modular Robotic Systems: Characteristics and Applications. Journal of Intelligence and Robotic Systems. Under press. 10.1007/s10846-015-0237-8.
[3] Chan, Kam Mahyuddin, Muhammad Khoo, Bee Ee. (2021). Network-Based Cooperative Synchronization Control of 3 Articulated Robotic Arms for Industry 4.0 Application. 10.1007/978-981-15-5281-630.
[4] Design of robot platform based on CAN bus.
[5] Economic 6-DOF robotic manipulator hardware design for research and education