Disclaimer

- The material provided in this document is not my original work and is a summary of some one else's work(s).
- A simple Google search of the title of the document will direct you to the original source of the material.
- I do not guarantee the accuracy, completeness, timeliness, validity, non-omission, merchantability or fitness of the contents of this document for any particular purpose.
-

CAN SCHEDULABILITY ANALYSIS REFUTED, REVISTED AND REVIEWED

Presented by

CAN SCHEDULABILITY ANALYSIS REFUTED, REVISTED AND REVIEWED

Presented by
나집 울라 칸
2013-4-11

OUTLINE

- Objective
- Introduction
- Response Time Analysis
- Priority Assignment Policies
- Implications of Flawed Analysis
- Conclusion

OBJECTIVE

OBJECTIVE

- We Discuss

OBJECTIVE

- We Discuss
  - Flaws in the analysis of CAN messages.

OBJECTIVE

- We Discuss
  - Flaws in the analysis of CAN messages.
  - Revised analysis resolving the problems with the original approach

OBJECTIVE

- We Discuss
  - Flaws in the analysis of CAN messages.
  - Revised analysis resolving the problems with the original approach
  - Optimality of priority assignment policies

OBJECTIVE

- We Discuss
  - Flaws in the analysis of CAN messages.
  - Revised analysis resolving the problems with the original approach
  - Optimality of priority assignment policies
  - Impact on commercial CAN systems designed and developed using flawed schedulability analysis

INTRODUCTION

INTRODUCTION

- CAN is

INTRODUCTION

- CAN is
  - Asynchronous

INTRODUCTION

- CAN is
  - Asynchronous
  - Multi-master

INTRODUCTION

- CAN is
  - Asynchronous
  - Multi-master
  - Serial data bus

INTRODUCTION

- CAN is
  - Asynchronous
  - Multi-master
  - Serial data bus
  - Priority based message arbitration(CSMA/CR)

INTRODUCTION

- CAN is
  - Asynchronous
  - Multi-master
  - Serial data bus
  - Priority based message arbitration(CSMA/CR)
  - Non-preemptive message transmission

INTRODUCTION

- CAN is
  - Asynchronous
  - Multi-master
  - Serial data bus
  - Priority based message arbitration(CSMA/CR)
  - Non-preemptive message transmission
  - Message IDs for ID and Priority

INTRODUCTION

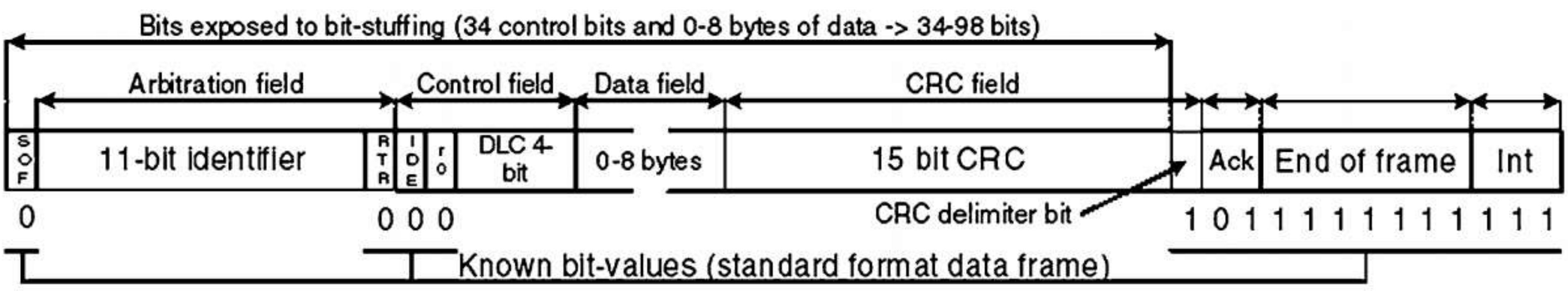
- CAN is
  - Asynchronous
  - Multi-master
  - Serial data bus
  - Priority based message arbitration(CSMA/CR)
  - Non-preemptive message transmission
  - Message IDs for ID and Priority
  - Dominant (‘0’) / Recessive (‘1’) bits

INTRODUCTION

- CAN is
  - Asynchronous
  - Multi-master
  - Serial data bus
  - Priority based message arbitration(CSMA/CR)
  - Non-preemptive message transmission
  - Message IDs for ID and Priority
  - Dominant (‘0’) / Recessive (‘1’) bits
  - Leading Edge of SOF for synchronization

INTRODUCTION

- CAN is
  - Asynchronous
  - Multi-master
  - Serial data bus
  - Priority based message arbitration(CSMA/CR)
  - Non-preemptive message transmission
  - Message IDs for ID and Priority
  - Dominant ('0') / Recessive ('1') bits



CONTD…

- g=34 for standard and g=54 for extended CAN

CONTD…

- Error Detection

  - g=34 for standard and g=54 for extended CAN

CONTD…

- Error Detection
  - Each message transmission that is signaled as an error can lead to a maximum of 31 additional bits 6 of error recovery overhead plus re-transmission of the message itself.

  - g=34 for standard and g=54 for extended CAN

CONTD…

- Error Detection
  - Each message transmission that is signaled as an error can lead to a maximum of 31 additional bits 6 of error recovery overhead plus re-transmission of the message itself.
- Bit Stuffing

  - g=34 for standard and g=54 for extended CAN

CONTD…

- Error Detection
  - Each message transmission that is signaled as an error can lead to a maximum of 31 additional bits 6 of error recovery overhead plus re-transmission of the message itself.
- Bit Stuffing
  - ‘000000’ and ‘111111’ are used to signal errors

  - g=34 for standard and g=54 for extended CAN

CONTD…

- Error Detection
  - Each message transmission that is signaled as an error can lead to a maximum of 31 additional bits 6 of error recovery overhead plus re-transmission of the message itself.
- Bit Stuffing
  - ‘000000’ and ‘111111’ are used to signal errors
  - Synchronization Problems

  - g=34 for standard and g=54 for extended CAN

CONTD…

- Error Detection
  - Each message transmission that is signaled as an error can lead to a maximum of 31 additional bits 6 of error recovery overhead plus re-transmission of the message itself.
- Bit Stuffing
  - ‘000000’ and ‘111111’ are used to signal errors
  - Synchronization Problems
  - CAN protocol therefore requires that a bit of the opposite polarity is inserted by the transmitter whenever 5 bits of the same polarity are transmitted

  - g=34 for standard and g=54 for extended CAN

CONTD…

- Error Detection
  - Each message transmission that is signaled as an error can lead to a maximum of 31 additional bits 6 of error recovery overhead plus re-transmission of the message itself.
- Bit Stuffing
  - ‘000000’ and ‘111111’ are used to signal errors
  - Synchronization Problems
  - CAN protocol therefore requires that a bit of the opposite polarity is inserted by the transmitter whenever 5 bits of the same polarity are transmitted

$$C_m = \left(g + 8s_m + 13 + \left\lfloor \frac{g + 8s_m - 1}{4} \right\rfloor \right) \tau_{\text{bit}}$$

  - g=34 for standard and g=54 for extended CAN

CONTD...

CONTD…

- Tindell et al showed how research into fixed priority pre-emptive scheduling for single processor systems could be adapted and applied to the scheduling of messages on CAN.

CONTD…

- Tindell et al showed how research into fixed priority pre-emptive scheduling for single processor systems could be adapted and applied to the scheduling of messages on CAN.
- This analysis provided a method of calculating the worst-case response times of all CAN messages

CONTD…

CONTD…

- Prior to Tindell's work low levels of bus utilization, up to 30 or 40%

CONTD…

- Prior to Tindell's work low levels of bus utilization, up to 30 or 40%
- With systematic approach to schedulability CAN bus utilization could be increased to around 80%.

CONTD…

- Prior to Tindell's work low levels of bus utilization, up to 30 or 40%
- With systematic approach to schedulability CAN bus utilization could be increased to around 80%.
- The design and development of many in vehicle Controller Area Networks relies on the schedulability analysis of CAN given by Tindell.

CONTD…

- Prior to Tindell’s work low levels of bus utilization, up to 30 or 40%
- With systematic approach to schedulability CAN bus utilization could be increased to around 80%.
- The design and development of many in vehicle Controller Area Networks relies on the schedulability analysis of CAN given by Tindell.
- Tindell’s work has been cited more than 200 times.

Flaws in Tindells work

FLAWS IN TINDELLS WORK

- The Tindells CAN Analysis has some flaws

FLAWS IN TINDELLS WORK

- The Tindells CAN Analysis has some flaws
- It may result in computed worst-case response times for messages that are optimistic, i.e. less than the response times that may actually occur.

CONTD…

CONTD…

- Lets illustrate this using an example

CONTD…

- Lets illustrate this using an example

Message	Priority	Period	Deadline	TX time
A	1	2.5 ms	2.5 ms	1 ms
B	2	3.5 ms	3.25 ms	1 ms
C	3	3.5 ms	3.25 ms	1 ms

CONTD…

- Lets illustrate this using an example

Message	Priority	Period	Deadline	TX time
A	1	2.5 ms	2.5 ms	1 ms
B	2	3.5 ms	3.25 ms	1 ms
C	3	3.5 ms	3.25 ms	1 ms

- Tindell’s analysis calculates the worst-case response times of messages A, B and C as 2 ms, 3 ms and 3 ms respectively.

CONTD…

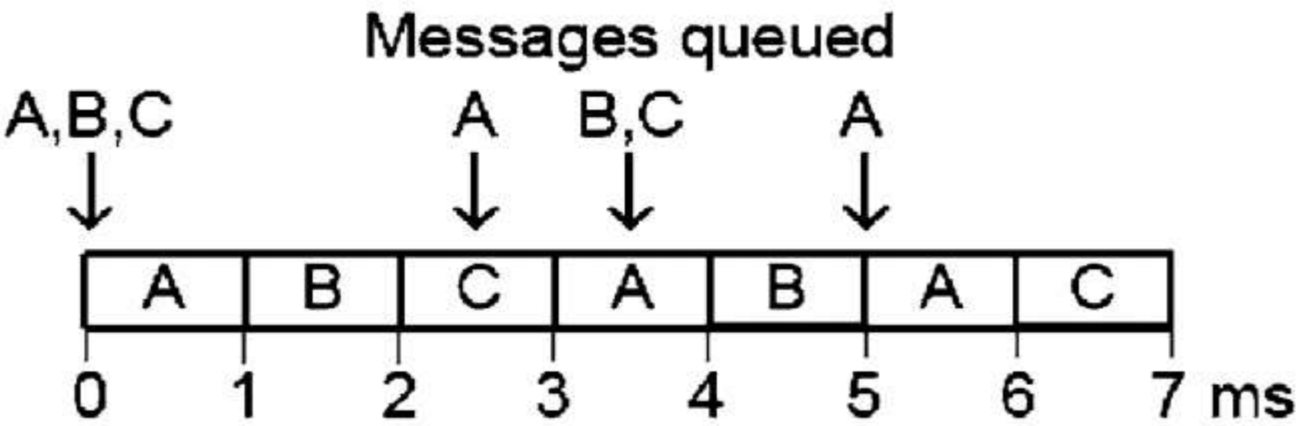
- Lets illustrate this using an example

Message	Priority	Period	Deadline	TX time
A	1	2.5 ms	2.5 ms	1 ms
B	2	3.5 ms	3.25 ms	1 ms
C	3	3.5 ms	3.25 ms	1 ms

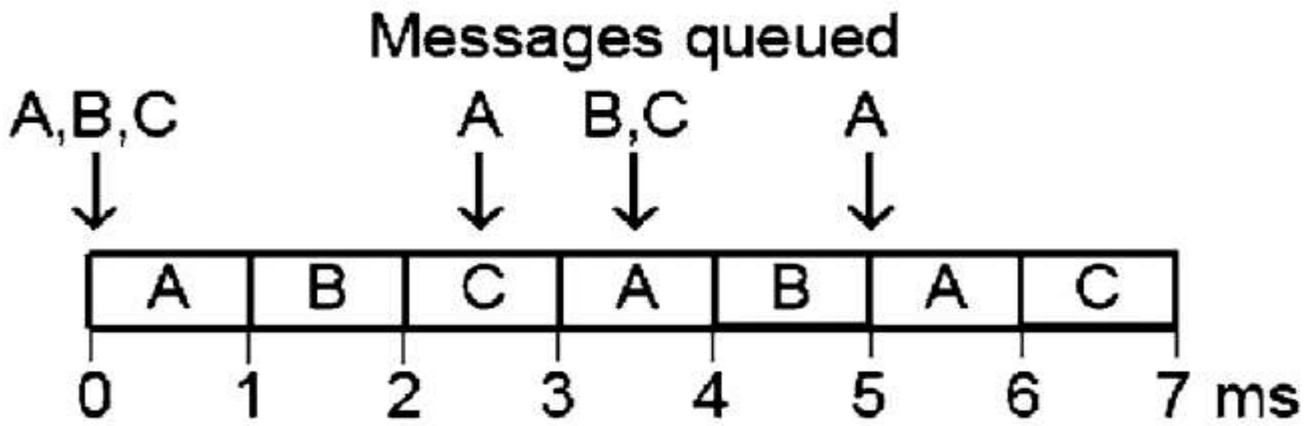
- Tindell's analysis calculates the worst-case response times of messages A, B and C as 2 ms, 3 ms and 3 ms respectively.
- Hence the system is deemed to be schedulable

CONTD…

CONTD…

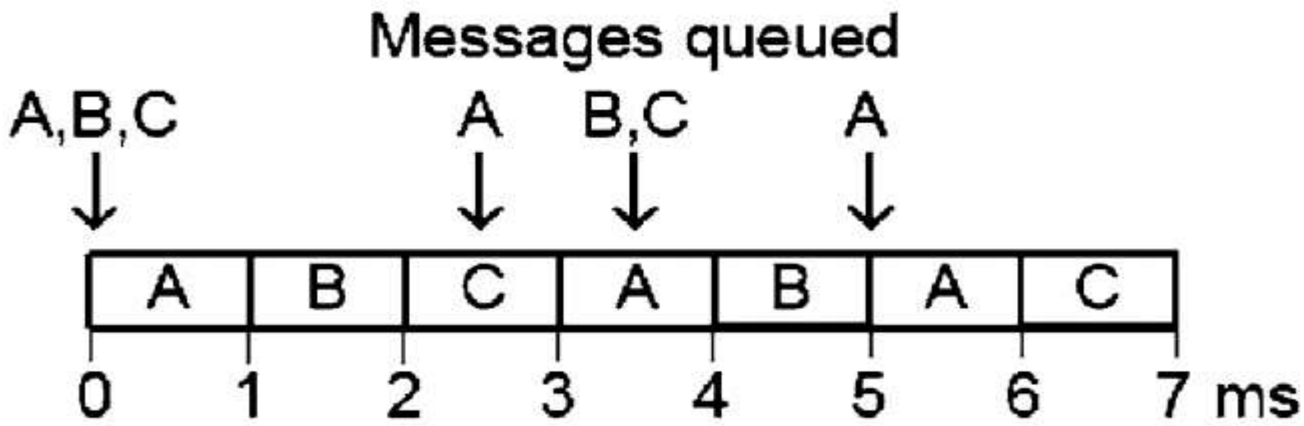


CONTD…



- Worst-case response time of C = 3ms according to Tindells Analysis

CONTD…



- Worst-case response time of C = 3ms according to Tindells Analysis
- However due to higher priority interference the actual worst-case response time for message C is 3.5 ms, which is greater than its deadline of 3.25 ms.

SCHEDULING MODEL

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.
- Some terminology

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.
- Some terminology
  - Maximum number of data bytes s_m

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.
- Some terminology
  - Maximum number of data bytes s_m
  - Maximum transmission time C_m

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.
- Some terminology
  - Maximum number of data bytes s_m
  - Maximum transmission time C_m
  - Queuing jitter J_m

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.
- Some terminology
  - Maximum number of data bytes s_m
  - Maximum transmission time C_m
  - Queuing jitter J_m
  - Minimum Inter-arrival time T_m

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.
- Some terminology
  - Maximum number of data bytes s_m
  - Maximum transmission time C_m
  - Queuing jitter J_m
  - Minimum Inter-arrival time T_m
  - Hard deadline D_m

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.
- Some terminology
  - Maximum number of data bytes s_m
  - Maximum transmission time C_m
  - Queuing jitter J_m
  - Minimum Inter-arrival time T_m
  - Hard deadline D_m
  - Worst-case response time R_m : longest time from the initiating event occurring to the message being received by the nodes that require it

SCHEDULING MODEL

- The system is assumed to comprise a number of nodes connected via CAN
- Each node is assumed to be capable of ensuring that, at any given time when arbitration starts, the highest priority message queued at that node is entered into arbitration.
- Some terminology
  - Maximum number of data bytes s_m
  - Maximum transmission time C_m
  - Queuing jitter J_m
  - Minimum Inter-arrival time T_m
  - Hard deadline D_m
  - Worst-case response time R_m : longest time from the initiating event occurring to the message being received by the nodes that require it
  - Schedulable $R_m <= D_m$

Response Time Analysis

RESPONSE TIME ANALYSIS

- RTA for CAN provide a method of calculating the worst-case response time of each message

RESPONSE TIME ANALYSIS

- RTA for CAN provide a method of calculating the worst-case response time of each message
- These values can then be compared to the message deadlines to determine if the system is schedulable.

RESPONSE TIME ANALYSIS

- RTA for CAN provide a method of calculating the worst-case response time of each message
- These values can then be compared to the message deadlines to determine if the system is schedulable.
- According to Tindell, the response time of a message can be viewed as being made up of three elements:

RESPONSE TIME ANALYSIS

- RTA for CAN provide a method of calculating the worst-case response time of each message
- These values can then be compared to the message deadlines to determine if the system is schedulable.
- According to Tindell, the response time of a message can be viewed as being made up of three elements:
  - Queuing jitter J_m

RESPONSE TIME ANALYSIS

- RTA for CAN provide a method of calculating the worst-case response time of each message
- These values can then be compared to the message deadlines to determine if the system is schedulable.
- According to Tindell, the response time of a message can be viewed as being made up of three elements:
  - Queuing jitter J_m
  - Queuing delay w_m

RESPONSE TIME ANALYSIS

- RTA for CAN provide a method of calculating the worst-case response time of each message
- These values can then be compared to the message deadlines to determine if the system is schedulable.
- According to Tindell, the response time of a message can be viewed as being made up of three elements:
  - Queuing jitter J_m
  - Queuing delay w_m
  - Transmission time C_m

RESPONSE TIME ANALYSIS

- RTA for CAN provide a method of calculating the worst-case response time of each message
- These values can then be compared to the message deadlines to determine if the system is schedulable.
- According to Tindell, the response time of a message can be viewed as being made up of three elements:
  - Queuing jitter J_m
  - Queuing delay w_m
  - Transmission time C_m
- Worst-case response time $R_m = J_m + w_m + C_m$

RTA CONTD…

- Maximum blocking time B_m

RTA CONTD…

- The Queuing delay w_m comprises two elements

- Maximum blocking time B_m

RTA CONTD…

- The Queuing delay w_m comprises two elements
  - Blocking B_m, due to lower priority messages which may be in the process of being transmitted when message m is queued

- Maximum blocking time B_m

RTA CONTD...

- The Queuing delay w_m comprises two elements
  - Blocking B_m, due to lower priority messages which may be in the process of being transmitted when message m is queued
  - Interference due to higher priority messages which may win arbitration and be transmitted in preference to message m
- Maximum blocking time B_m

RTA CONTD…

- The Queuing delay w_m comprises two elements
  - Blocking B_m, due to lower priority messages which may be in the process of being transmitted when message m is queued
  - Interference due to higher priority messages which may win arbitration and be transmitted in preference to message m
- Maximum blocking time B_m

  - Where lp(m) is the set of messages with lower priority than m

RTA CONTD…

- The Queuing delay w_m comprises two elements
  - Blocking B_m, due to lower priority messages which may be in the process of being transmitted when message m is queued
  - Interference due to higher priority messages which may win arbitration and be transmitted in preference to message m
- Maximum blocking time B_m

$$B_m = \max_{k \in lp(m)} (C_k)$$

  - Where lp(m) is the set of messages with lower priority than m

RTA CONTD…

RTA CONTD…

- Priority Level-m Busy Period

RTA CONTD…

- Priority Level-m Busy Period
  - It starts at some time t^s when a message of priority m or higher is queued ready for transmission and there are no messages of priority m or higher waiting to be transmitted that were queued strictly before time t^s

RTA CONTD…

- Priority Level-m Busy Period
  - It starts at some time t^s when a message of priority m or higher is queued ready for transmission and there are no messages of priority m or higher waiting to be transmitted that were queued strictly before time t^s
  - It is a contiguous interval of time during which any message of priority lower than m is unable to start transmission and win arbitration.

RTA CONTD…

- Priority Level-m Busy Period
  - It starts at some time t^s when a message of priority m or higher is queued ready for transmission and there are no messages of priority m or higher waiting to be transmitted that were queued strictly before time t^s
  - It is a contiguous interval of time during which any message of priority lower than m is unable to start transmission and win arbitration.
  - It ends at the earliest time t^e when the bus becomes idle, ready for the next round of transmission and arbitration, yet there are no messages of priority m or higher waiting to be transmitted that were queued strictly before time t^e

RTA CONTD…

RTA CONTD…

- All messages of priority m or higher, queued strictly before the end of the busy period, are transmitted during the busy period.

RTA CONTD…

- All messages of priority m or higher, queued strictly before the end of the busy period, are transmitted during the busy period.
- These messages cannot therefore cause any interference on a subsequent instance of message m queued at or after the end of the busy period

RTA CONTD...

- All messages of priority m or higher, queued strictly before the end of the busy period, are transmitted during the busy period.
- These messages cannot therefore cause any interference on a subsequent instance of message m queued at or after the end of the busy period
- Critical Instant: where the maximal busy period begins and message m is queued simultaneously with all higher priority messages

RTA CONTD…

- The worst case queuing delay is given by

$$w_m = B_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m + J_k + \tau_{\mathrm{bit}}}{T_k} \right\rceil C_k$$

- The worst-case response time of the first Instance of the message in the busy period is given by: $J_m + w_m^{n+1} + C_m$
- The flaw in this analysis is that, given the constraint $D_m \leq T_m$, it implicitly assumes that if message m is schedulable, then the priority level-m busy period will end at or before T_m.

RTA CONTD…

RTA CONTD…

- With fixed priority non-preemptive scheduling, a higher priority message can be awaiting transmission when message m completes transmission, and thus the busy period can extend beyond T_m.

RTA CONTD…

- With fixed priority non-preemptive scheduling, a higher priority message can be awaiting transmission when message m completes transmission, and thus the busy period can extend beyond T_m.
- The length t_m, of the priority level-m busy period is given by the following recurrence relation

RTA CONTD...

- With fixed priority non-preemptive scheduling, a higher priority message can be awaiting transmission when message m completes transmission, and thus the busy period can extend beyond T_m.
- The length t_m, of the priority level-m busy period is given by the following recurrence relation

$$t_m^{n+1} = B_m + \sum_{\forall k \in hep(m)} \left\lceil \frac{t_m^n + J_k}{T_k} \right\rceil C_k$$

RTA CONTD…

RTA CONTD…

- If $t_m \leq T_m - J_m$, then the busy period ends at or before the second instance of message m is queued.

RTA CONTD…

- If $t_m \leq T_m - J_m$, then the busy period ends at or before the second instance of message m is queued.
- If $t_m > T_m - J_m$, then the existing analysis may give an optimistic worst-case response time, depending on whether the first, or a subsequent instance of message m has the longest response time.

RTA CONTD…

- The number of instances Q_m, of message m that become ready for transmission before the end of the busy period is given by

$$Q_m = \left\lceil \frac{t_m + J_m}{T_m} \right\rceil$$

- To determine the worst-case response time of message m, it is necessary to calculate the response time of each of the Q_m instances
- The maximum of these values then gives the worst-case response time.

RTA CONTD…

- The longest time from the start of the busy period to instance q beginning successful transmission is given by

$$w_m^{n+1}(q) = B_m + qC_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{\text{bit}}}{T_k} \right\rceil C_k$$

- The event initiating instance q of the message occurs at time $qT_m - J_m$ so

$$R_m(q) = J_m + w_m(q) - qT_m + C_m$$

RTA CONTD…

- The worst-case response time of message m is therefore

$$R_m = \max_{q=0..Q_m-1} (R_m(q))$$

- If $D_m > T_m$ then CAN controller hardware may need to be capable of buffering more than one instance of a message
- N_m the number of instances of each message that need to be buffered is bounded by

$$N_m = \left\lceil \frac{R_m}{T_m} \right\rceil$$

RTA CONTD…

- Example:

RTA CONTD…

- Example:

Message	Priority	Period	Deadline	TX time
A	1	2.5 ms	2.5 ms	1 ms
B	2	3.5 ms	3.25 ms	1 ms
C	3	3.5 ms	3.25 ms	1 ms

RTA CONTD…

- Example:

RTA CONTD...

- Example:

$$t_m^{n+1} = B_m + \sum_{\forall k \in hep(m)} \left\lceil \frac{t_m^n + J_k}{T_k} \right\rceil C_k$$

RTA CONTD…

- Example:

$$t_m^{n+1} = B_m + \sum_{\forall k \in hep(m)} \left\lceil \frac{t_m^n + J_k}{T_k} \right\rceil C_k$$

a value of $t_3^0 = C_3 = 1$, the recurrence relation given by Eq. (8) iterates as follows: $t_3^1 = 3$, $t_3^2 = 4$, $t_3^3 = 6$, $t_3^4 = 7$, converging as $t_3^5 = t_3^4 = 7$. The length of the busy period is therefore 7.0 ms, and the number of instances of message C that need to be examined is given by Eq. (10):

$$Q_3 = \left\lceil \frac{7.0}{3.5} \right\rceil = 2$$

RTA CONTD…

- Example:

RTA CONTD…

$$w_m^{n+1}(q) = B_m + qC_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{\text{bit}}}{T_k} \right\rceil C_k$$

RTA CONTD…

$$w_m^{n+1}(q) = B_m + qC_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{\text{bit}}}{T_k} \right\rceil C_k$$

Calculation of the response time of the first instance proceeds using Eq. (11): $w_3^0(0) = 0$, $w_3^1(0) = 2$, converging as $w_3^2(0) = w_3^1(0) = 2$. Using Eq. (12), we have $R_3(0) = 3$, the same response time calculated by the existing analysis.

Moving on to the second instance, $w_3^0(1) = w_3(0) + C_m = 3$, $w_3^1(1) = 4$, $w_3^2(1) = 5$, $w_3^3(1) = 6$. At this point computation would normally stop as the response time, given by $J_3 + w_3(q) - qT_3 + C_3$, has reached 3.5 ms which is greater than the message deadline. However, if we continue iterating, assuming a longer deadline, then the recurrence relation converges on $w_3^4(1) = w_3^3(1) = 6$ and hence $R_3(1) = 3.5$ ms. The worst-case response time of message C is in fact 3.5 ms, as previously illustrated

RTA CONTD…

- Sufficient Schedulability Tests
- The schedulability tests given in this section are only applicable given the constraint that message deadlines do not exceed their periods.
- Assuming that this first instance completes transmission before its deadline and hence before the end of its period, then we have two possibilities to consider.
  - Busy period ends before the next instance of message m is queued
  - Busy period continues beyond the time at which the next instance of message m is queued

CONTD…

- $s_{m,q}$ and $s_{m,q+1}$, at which two arbitrary but consecutive instances q and q+1, of message m start transmission.
- An upper bound on the length of the time interval $[s_{m,q}, s_{m,q+1})$ can be found by making the potentially pessimistic assumption that all higher priority messages are queued just as instance q starts transmission

$$w_m^{n+1} = C_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{\text{bit}}}{T_k} \right\rceil C_k$$

CONTD…

- Given the assumption that the first q instances in the busy period are schedulable, and the constraint that $D_m \leq T_m$, then the start (and end) of transmission of the qth instance must happen before the end of its period, and hence before the (q+1)th instance is queued.
- The queuing delay for the (q+1)th instance, as measured from the time at which it is queued to the start of its transmission, is less than the length of the interval $[s_{m,q}, s_{m,q+1})$.

CONTD…

- Intuitively, the second and subsequence instances of message m in the busy period are subject to blocking, of at most C_m, due to the previous instance of the same message.

$$w_m^{n+1} = \max(B_m, C_m) + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{\text{bit}}}{T_k} \right\rceil C_k$$

- An instance of message m can be subject to blocking; either of B_m, due to non-preemptive transmission of lower priority messages; or of Cm, due to the non-pre-emptive transmission of the previous instance of message m itself.

CONTD…

- Further Simplifying

$$w_m^{n+1} = B^{\mathrm{MAX}} + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{\mathrm{bit}}}{T_k} \right\rceil C_k$$

ERROR MODEL

ERROR MODEL

- We assume that the maximum number of errors present on the bus in some time interval t is given by the function F(t)

ERROR MODEL

- We assume that the maximum number of errors present on the bus in some time interval t is given by the function F(t)
- The worst-case impact of a single bit error is to cause transmission of an additional 31 bits of error recovery overhead plus re-transmission of the affected message.

ERROR MODEL

- We assume that the maximum number of errors present on the bus in some time interval t is given by the function F(t)
- The worst-case impact of a single bit error is to cause transmission of an additional 31 bits of error recovery overhead plus re-transmission of the affected message.
- Only errors affecting message m or higher priority messages can delay message m from being successfully transmitted

ERROR MODEL

- We assume that the maximum number of errors present on the bus in some time interval t is given by the function F(t)
- The worst-case impact of a single bit error is to cause transmission of an additional 31 bits of error recovery overhead plus re-transmission of the affected message.
- Only errors affecting message m or higher priority messages can delay message m from being successfully transmitted

$$E_m(t) = \left(31\tau_{\text{bit}} + \max_{k \in hep(m)} (C_k)\right) F(t)$$

CONTD…

- Busy Period

- Worst Case Response Time

CONTD...

- Busy Period

$$t_m^{n+1} = E_m\left(t_m^n\right) + B_m + \sum_{\forall k \in hep(m)} \left\lceil \frac{t_m^n + J_k}{T_k} \right\rceil C_k$$

- Worst Case Response Time

CONTD…

- Busy Period

$$t_m^{n+1} = E_m\left(t_m^n\right) + B_m + \sum_{\forall k \in hep(m)} \left\lceil \frac{t_m^n + J_k}{T_k} \right\rceil C_k$$

- Worst Case Response Time

$$w_m^{n+1}(q) = E_m\left(w_m^n + C_m\right) + B_m + qC_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{\text{bit}}}{T_k} \right\rceil C_k$$

CONCLUSIONS FROM DISCUSSION

- Any message from the lowest priority to the 3rd highest priority in a set of 3 or more messages can be given an optimistic response time and therefore a faulty guarantee by the existing analysis
- The above statement is true even for low utilization levels.
- If the existing analysis showed that every message was schedulable in the presence of any reasonable error model, with $F(t) \geq 1$, then, despite the flaw in the existing analysis, every message is actually guaranteed to be schedulable when no errors are present.

CONTD…

- The omission of a single maximum length message of arbitrary priority provides sufficient reduction in interference/blocking to ensure that the flaw in the existing analysis cannot lead to any of the remaining messages missing their deadlines.
- If $B_m \geq C_m$, then the first instance of message m is guaranteed to have a response time at least as long as subsequent ones.

Priority Assignment Policies

PRIORITY ASSIGNMENT POLICIES

- Tindell claimed that deadline monotonic priority assignment was optimal for CAN.

PRIORITY ASSIGNMENT POLICIES

- Tindell claimed that deadline monotonic priority assignment was optimal for CAN.
- However its not, consider an example

PRIORITY ASSIGNMENT POLICIES

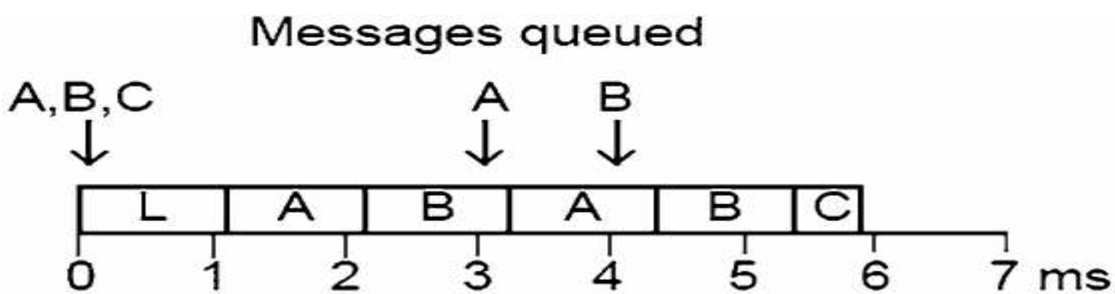
- Tindell claimed that deadline monotonic priority assignment was optimal for CAN.
- However its not, consider an example

Message	Period	Deadline	Number of bits	TX time
A	3.0 ms	3.0 ms	135	1.08 ms
B	4.0 ms	4.0 ms	135	1.08 ms
C	4.5 ms	4.5 ms	65	0.52 ms

PRIORITY ASSIGNMENT POLICIES

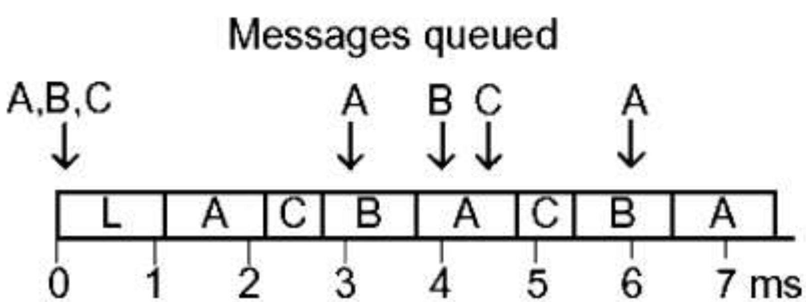
- Tindell claimed that deadline monotonic priority assignment was optimal for CAN.
- However its not, consider an example

Message	Period	Deadline	Number of bits	TX time
A	3.0 ms	3.0 ms	135	1.08 ms
B	4.0 ms	4.0 ms	135	1.08 ms
C	4.5 ms	4.5 ms	65	0.52 ms



CONTD…

- If we use the priority ordering A, C, B then the worst-case response times of the messages are: RA=2.16 ms, RC=2.68 ms and RB=3.76 ms



OPTIMAL PRIORITY ASSIGNMENT

```
Optimal Priority Assignment Algorithm

for each priority level, lowest first
{
   for each unassigned message m
   {
       if m is schedulable at this priority
       {
           assign m this priority
           break (continue outer loop)
       }
   }
    return unschedulable
}
return schedulable
```

CONCLUSION

1. The existing analysis can provide optimistic worst-case response times for messages from the 3rd highest priority to the lowest priority.
2. The existing analysis can lead to broken guarantees and hence deadline misses in systems with low bus utilisation.
3. Where an error model has been considered, the flaw in the existing analysis is not sufficient to lead to CAN configurations that will result in missed deadlines when no errors are present on the bus. The desired robustness to errors may not however be achieved.
4. The omission of a single maximum length diagnostic message, accounted for by the existing analysis, reduces interference/blocking enough to ensure that the deadlines of all the remaining messages are met during normal operation.
5. Despite its flaws, the existing analysis gives the correct response time for any message where there is at least one lower priority message with the same or longer transmission time/message length.