Projecto 1 - Aconteceu... em Marte

Luís Ferreirinha
Nº51127
Mestrado em Informática

Rodrigo Branco
Nº54457
Mestrado em Engenharia Informática

***Abstract*—Neste projeto utilizou-se a ferramenta Cheddar para simular e analisar o escalonamento do sistema utilizado pela sonda "Mars Pathfinder". Recriou-se históricamente o sistema, fazendo uma descrição detalhada a partir do qual se determinou quais as tarefas e recursos mais relevantes para descrever o sistema com precisão e analisou-se do ponto de vista do escalonamento. Foram considerados 3 diferentes cenários, os quais foram analisados e comentados em relação ás suas especificações e fatores determinantes que permitem o escalonamento nos respetivos ambientes. Comentou-se um quarto cenário onde foi dada uma opinião e deu-se um ponto de vista em relação ás melhores definições do sistema utilizado sem ter em conta a era em que tal evento aconteceu. Por fim procedeu-se a um breve análise sobre alguns desafios e dificuldades encontradas ao realizar o projeto.**

I. Introdução

O Mars Pathfinder é uma sonda espacial que foi lançada pela NASA em 1996, com destino ao planeta Marte. Esta aterrou no dia 4 de Julho de 1997 na superfície do planeta, e transportava como payload uma base station e o rover Sojourner. Ambos os equipamentos traziam um conjunto de diferentes instrumentos científicos, tendo como objetivo a análise da atmosfera, clima e geologia Marciana. Mais especificamente a base station do Pathfinder lander vinha equipada com o seguinte: Imager for Mars Pathfinder (IMP), e Atmospheric and meteorological sensors (ASI/MET).

Alguns dias após o inicio da missão, o PathFinder começou a recolher dados atmosféricos através do instrumento ASI/MET. A operação da sonda parecia normal até esta começar subitamente a experienciar system resets totais, causando perdas de dados. Após uma investigação por parte dos engenheiros numa réplica da sonda, foi descoberto que os system resets aconteciam devido a um deadline miss da task bc_dist. Esta deadline miss ocorria devido a uma inversão de prioridades, pois a task de maior prioridade bc_dist era bloqueada por uma task de prioridade muito menor ASI/MET que tinha adquirido uma lock para um recurso partilhado. A task ASI/MET depois de adquirir este lock era interrompida por várias tasks de prioridade média. Esta interrupção fazia com que a task ASI/MET não libertasse o lock, por consequente, a task bc_dist que necessitava do lock acabava por não executar, no fim de um período a task bc_sched verificava se as tarefas tinham sido concluídas no fim do período (125 ms), como tal não acontecia reiniciava o sistema. Este ciclo pode ser repetido indefinidamente.

O objetivo deste trabalho então é realizar uma recriação histórica do funcionamento da sonda Mars Pathfinder e analizar o seu funcionamento. Para tal vão ser recriados 3 cenários diferentes: Um primeiro no período logo após o contacto com a superfície de Marte, um segundo apartir do momento em que se inicia a atividade de recolha de dados científicos e meteorológicos e um terceiro onde se aplica a solução proposta e implementada pelos engenheiros da missão ao problema da inversão de prioridades. Por último, serão analisadas outras soluções para o problema, mesmo que estas não fossem possíveis de aplicar nesta situação.

II. Descrição do sistema

A. Hardware e Arquitetura

A sonda Mars PathFinder, como foi mencionado na introdução, é consituida por uma base station com dois instrumento científicos:

1) Imager for Mars Pathfinder (IMP)
2) Atmospheric and meteorological sensors (ASI/MET)

O computador presente na sonda seria constituído por um Radiation Hardened IBM Risc 6000 Single Chip (Rad6000 SC) CPU com 128 MB de Ram e 6 MB de EEPROM e corria o sistema operativo VxWorks.

O CPU residiria num VME bus que contem placas de interface para o rádio, câmara e um 1553 bus. O 1553 bus está ligado a dois sítios da sonda: A "cruise stage" e a parte "lander". Para esta recriação, estamos apenas interessados nos instrumentos ligados à parte "lander" da sonda, que são um Altímetro Rádio, um Acelerómetro e o ASI/MET. Na figura 1 podemos encontrar um diagrama aproximado da arquitetura da sonda.

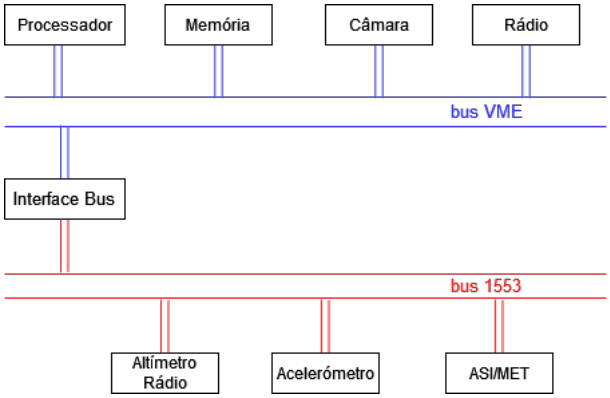

Fig. 1. Diagrama aproximado da arquitetura da Sonda PathFinder

B. Tasks do Sistema

O Hardware utilizado para realizar interface com o 1553 bus tem um paradigma específico de utilização, requerendo que o software efetue o escalonamento das tasks com uma frequência de 8Hz (a cada 125ms).

O software utilizado para controlar o 1553 bus e os instrumentos ligados foi implementado utilizando duas tasks. O bc_sched (bus scheduler) que realiza o setup das transações no bus e o bc_dist (distribution) que estava encarregue de recolher os resultados das transações. Estas duas tarefas verificam se a concluí-o a sua execução em cada ciclo.

De acordo com [1] as tasks do PathFinder têm a seguinte hierarquia de prioridades:

1) bc_sched
2) A task controlling the entry and landing
3) bc_dist
4) Tasks which perform other spacecraft functions
5) Science functions, such as image compression, and the ASI/MET

Como não conhecemos todas as tasks que são executadas pelo computador da sonda PathFinder vamos ter fazer algumas assunções sobre as restantes tarefas, mas utilizando a hierarquia definida anteriormente e o diagrama aproximado da arquitetura podemos induzir a existência de algumas tasks. Nomeadamente teríamos pelo menos uma task para cada uma das componentes presentes no diagrama (sem contar com o processador e memória), portanto vamos considerar as seguintes tasks adicionais: COMM (Task de operação do rádio), CAM (task de operação da câmara), ALT (task de leitura altímetro), ACCE (leitura do acelerómetro) e por último a ASI/MET (leitura dos sensores metereológicos). Para cada uma destas tasks vamos ter de atribuir prioridades, com base na hierarquia já definida já sabemos que ASI/MET é de baixa prioridade, enquanto que para as outras vamos ter de primeiro as classificar como sendo ”other spacecraft functions” ou ”Science functions”. As tasks de controlo de entrada e aterragem não são consideradas pois estamos a recriar situações após a aterragem da sonda, onde supomos que tais tarefas deixem de ser executadas devido aos segmentos encarregues da aterragem já terem terminado a sua missão. Portanto vamos considerar as tasks COMM e CAM como sendo ”other spacecraft functions”, tendo assim uma prioridade intermédia, e as tasks ACCE e ALT como sendo ”Science functions” com uma prioridade baixa, finalmente a task ASI/MET é frequentemente descrita como sendo uma das tasks de menor prioridade portanto decidimos colocar esta como a menor prioridade de todas.

Com as tasks definidas temos agora de determinar os parâmetros destas. Começando pelas tasks COMM e CAM, consideramos que estas tasks vão ter uma duração maior e portanto determinamos que ambas terão um tempo de execução de 75 (unidades de tempo do Cheddar), como estas não são tasks cruciais ao funcionamento sistema e devido ao seu tempo de execução maior decidimos que estas devem ser executadas a cada 500 unidades de tempo e que devem de ter uma Meta igual ao período. Para as tasks SENSOR_ALT e SENSOR_ACCE consideramos que ambas têm uma curta duração de 40 unidades de tempo, pois apenas leem valores de sensores mais simples, e que são bastante infrequentes com um período e meta de 1000 unidades de tempo. Finalmente para a task ASI/MET decidimos colocar um tempo de execução de 45 unidades, ligeiramente maior que os outros sensores, e como esta é descrita como uma atividade infrequente colocamos um período e meta de 1000 unidades. Os offsets de cada task foram depois ajustados manualmente de forma a recriar os cenários pedidos. Resumimos as nossas assunções na tabela I.

C. Recursos Partilhados

Quanto aos recursos do sistema como podemos ver pelo diagrama da Figura 1, existem dois bus o VME e o 1553. Sabemos por [1], [2] e [3] que o problema que levou ao reset do sistema foi devido ao uso do bus 1553, tendo sido explicitamente indicado por Glenn Reeves em [1] que as tasks que tinham acesso ao bus 1553 acediam a este através de um mecanismo de double buffered shared memory com a exeção da task ASI/MET que entregava a sua informação através de IPC (Inter Process Comunication). O mecanismo de IPC utiliza o pipe() do VxWorks para enviar mensagens pelo bus, as tasks que estão à espera de uma mensagem enviada por este meio utilizam o mecanismo select() para esperar pela chegada da mensagem. O mecanismo select() cria um semáforo de exclusão mútua de forma a proteger a lista de espera das mensagens e ao utilizar este mutex a task bloqueia o acesso ao bus 1553 através deste mesmo mecanismo de IPC.

De acordo com estes detalhes decidimos que para recriar o sistema da forma mais fidedigna, devíamos apenas modelar o bus 1553 e considerar o uso deste apenas por parte das tasks bc_dist e ASI/MET. Sabemos que no diagrama da figura 1, estão duas componentes extra a utilizar este bus e que

TABLE I
PARÂMETROS UTILIZADOS NA MODELAÇÃO EM CHEDDAR DE CADA TAREFA CONSIDERADA PARA EXECUÇÃO NO SISTEMA DA SONDA PATHFINDER.

Tarefa	**Prioridade**	**Período** (P_i)	**Meta** (D_i)	**Offset**	**Tempo Execução** (C_i)
BC_SCHD	8	125	125	100	20
BC_DIST	6	125	100	55	35
COMM	5	500	500	280	75
CAM	4	500	500	50	75
SENSOR_ALT	2	1000	1000	20	40
SENSOR_ACCE	2	1000	1000	500	40
ASI/MET	1	1000	1000	250	45

TABLE II
INTERVALOS DE UTILIZAÇÃO DO RECURSO BUS 1553 POR PARTE DAS TAREFAS BC_DIST E ASI/MET.

Tarefa	Aquisição do Recurso	Libertação do Recurso
BC_DIST	2	35
ASI/MET	10	40

modelamos as tasks (SENSOR_ACCE e SENSOR_ALT) para cada uma destas. Mas de acordo com [1], estas tasks não utilizam o mecanismo de IPC para comunicar por este bus, mas sim o mecanismo de double buffered shared memory, portanto as suas interações com o recurso não serão iguais às do ASI/MET e bc_dist, e adicionalmente não parecem causar conflitos ao utilizar este bus. Em relação às componentes que comunicam pelo bus VME, também não temos informação sobre o método utilizado para realizar a comunicação e adicionalmente não temos indicação por nenhuma das fontes de problemas relacionados com este bus, portanto decidimos não modelar este recurso.

Finalmente em relação ao tempo de utilização dos recursos pelas tasks, decidimos que a task ASI/MET primeiro precisa de recolher alguns dados antes de os enviar portanto colocamos a sua utilização a começar após 10 unidades de tempo e terminar após 40 unidades de tempo de execução da task. Para a task BC_DIST como esta está encarregue de gerir as coleção dos dados enviados pelo bus, decidimos colocar a sua utilização a começar após 2 unidades e a terminar quando esta task termina. Resumimos isto na tabela II.

D. Modelação do Sistema no Cheddar

Com a nossa descrição do sistema da sonda PathFinder concluída, vamos agora especificar os componentes que vamos modelar no software Cheddar:

1) CPU: O Processador RAD6000 foi modelado como um Monocore com velocidade de 1 e utilizando o scheduler Posix 1003 Highest Priority First Protocol

2) Tasks: Foram modeladas as seguintes tasks do sistema BC_SCHD, BC_DIST, COMM, CAM, SENSOR_ALT, SENSOR_ACCE e ASI/MET. Os parâmetros utilizados para cada uma destas podem ser encontrados na tabela I.

3) Resources: O bus 1553 foi modelado como um recurso com estado 1 que é utilizado pelas tasks ASI/MET e BC_DIST. O protocolo utilizado depende do cenário, sendo indicado nas secções seguintes. E o tempo de utilização por cada task pode ser encontrado na tabela II.

III. RECRIAÇÃO HISTÓRICA

Baseado-nos na descrição anterior do sistema, vamos agora recriar e analisar 4 cenários diferentes. Os cenários que vamos considerar são os seguintes:

- Cenário 1 - Operação da Sonda Após Contacto com a Superfície de Marte
- Cenário 2 - Operação da Sonda Após a Ativação da Tarefa ASI/MET
- Cenário 3 – Resolução do Problema – Mecanismos de Herança de Prioridades
- Cenário 4 – Mecanismos de Teto de Prioridades

Sendo os cenários 1-3 uma recriação dos eventos reais que decorreram em Marte, e o quarto uma situação hipotética em que se considera como solução um mecanismo não existente no núcleo de sistema operativo no computador da sonda PathFinder.

A. Cenário 1

No primeiro cenário implementámos as tarefas correspondentes na tabela à exceção da tarefa ASI/MET como explicitado no enunciado, e introduzimos o 1553 BUS como um recurso partilhado usado unicamente pela tarefa BC_DIST. De seguida realizamos uma simulação do escalonamento até 1000 unidades de tempo com o software Cheddar. Encontramos que para este cenário todas as tasks que adicionamos no Cheddar, com os respetivos parâmetros mencionadas na secção anterior, são escalonáveis, tendo que obtivemos os resultados de worst task response time demonstrados na tabela III e obtivemos também o diagrama temporal da figura 2.

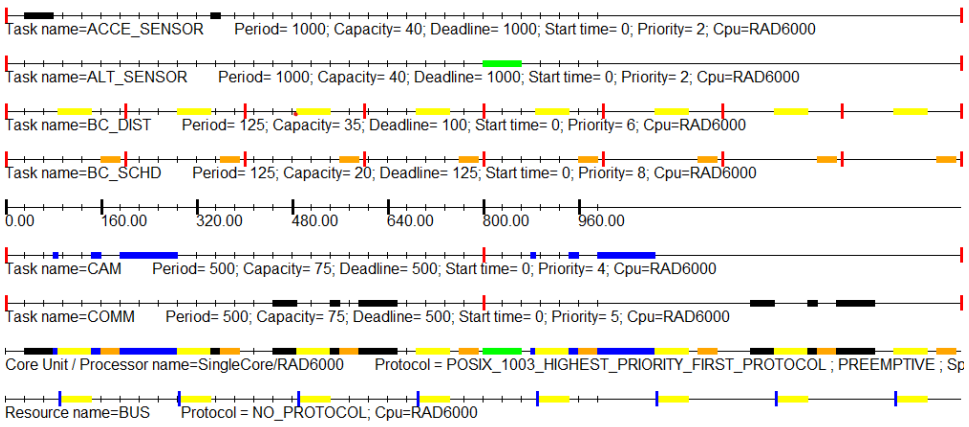

Fig. 2. Diagrama Temporal da simulação no Cheddar do escalonamento das tarefas no cenário 1 durante 1000 unidades de tempo.

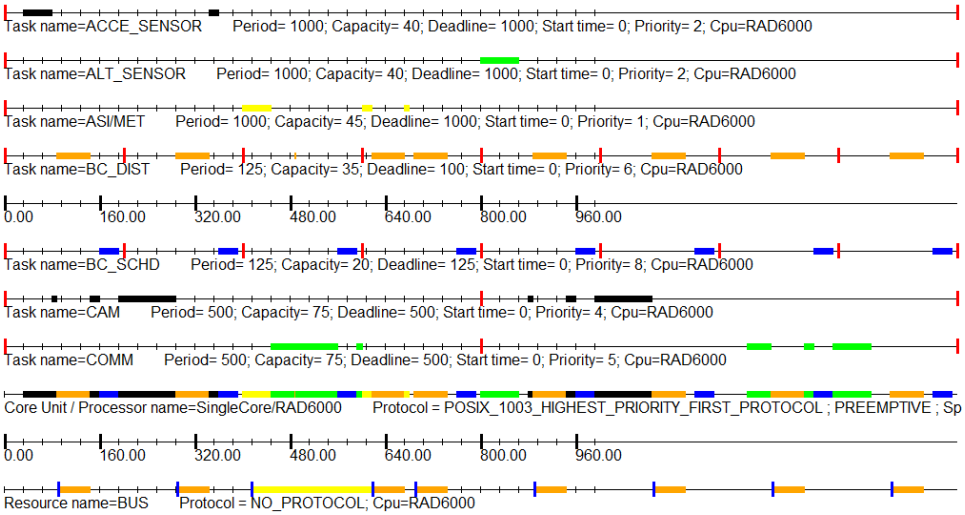

Fig. 3. Diagrama Temporal da simulação no Cheddar do escalonamento das tarefas no cenário 2 durante 1000 unidades de tempo.

TABLE III
WORST TASK RESPONSE TIME CENÁRIO 1

Tarefa	WTRT
ACCE_SENSOR	255
ALT_SENSOR	540
BC_DIST	90
BC_SCHD	120
CAM	180
COMM	410

B. Cenário 2

No segundo cenário como explicitado no enunciado adicionámos a tarefa ASI/MET, esta tarefa foi colocada a partilhar o recurso BUS 1553 com o BC_DIST, e como no cenário anterior realizamos uma simulação do escalonamento das tasks para 1000 unidades de tempo utilizando o Cheddar. Esta simulação indicou-nos que as tarefas não vão ser escalonáveis (tal como aconteceu na sonda PathFinder) e vamos observar pela análise do worst task response time na tabela IV e o diagrama temporal da figura 3, que isto deve-se a um incumprimento da deadline da tarefa BC_DIST.

Se analisarmos com atenção o diagrama da figura IV vamos notar que após a segunda deadline da task BC_DIST (task laranja no diagrama), a task ASI/MET (task amarela no diagrama) começa a sua execução adquirindo a lock do BUS 1553 sendo de seguida interrompida pela task de média prioriade COMM (segunda task verde no diagrama). Quando esta task é interrompida ela continua com a mutex lock do recurso, como podemos ver pelo uso do recurso BUS no diagrama, então quando a task BC_DIST tenta de seguida executar não consegue adquirir a lock do recurso e é incapaz de proseguir com a sua execução resultando na sua interrupção por outra task, nest caso a COMM. A task BC_DIST só é capaz de continuar a sua execução depois de a task ASI/MET terminar e libertar a lock do recurso. Todo este processo causa com a que task BC_DIST falhe a sua deadline, e representa o problema de inversão de prioridades descoberto na sonda PathFinder.

TABLE IV
WORST TASK RESPONSE TIME CENÁRIO 2

Tarefa	WTRT
ACCE_SENSOR	225
ALT_SENSOR	540
BC_DIST	170
BC_SCHD	120
CAM	180
COMM	410
ASI/MET	425

Se utilizarmos a ferramenta: ”Tools -> Resource -> Looking For Priority Inversion from Simulation” podemos obter a informação se que houve uma mudança de prioridade onde BC_DIST corre depois de uma tarefa de prioridade mais baixa no período indicado (de 306 a 386).

C. Cenário 3

Neste cenário aplicámos a mesma abordagem utilizada pelos engenheiros da missão Mars PathFinder, aplicar o protocolo Priority Inheritance Protocol (PIP) à utilização do recuso BUS 1553. De forma análoga aos cenários anteriores, realizamos uma simulação do escalonamento para 1000 unidades de tempo, utilizando o Cheddar. Esta abordagem, como podemos observar na tabela V e no diagrama da figura 4, resolveu o problema da deadline miss da tarefa BC_DIST, agora o pior caso da tarefa é 100 unidades de tempo, o que é aceite já que a deadline definida para este é 100.

O protocolo PIP permite resolver o problema da inversão de prioridades neste caso porque, quando a task BC_DIST necessita do recurso e a task ASI/MET tem a lock deste recurso e foi interrompida por outra tarefa de prioridade média (COMM), a task ASI/MET herda a prioridade da task de maior prioridade que precisa do recurso (BC_DIST) e então interrompe a task de prioridade média que a tinha previamente interrompido (COMM) e executa até libertar a lock do recurso.

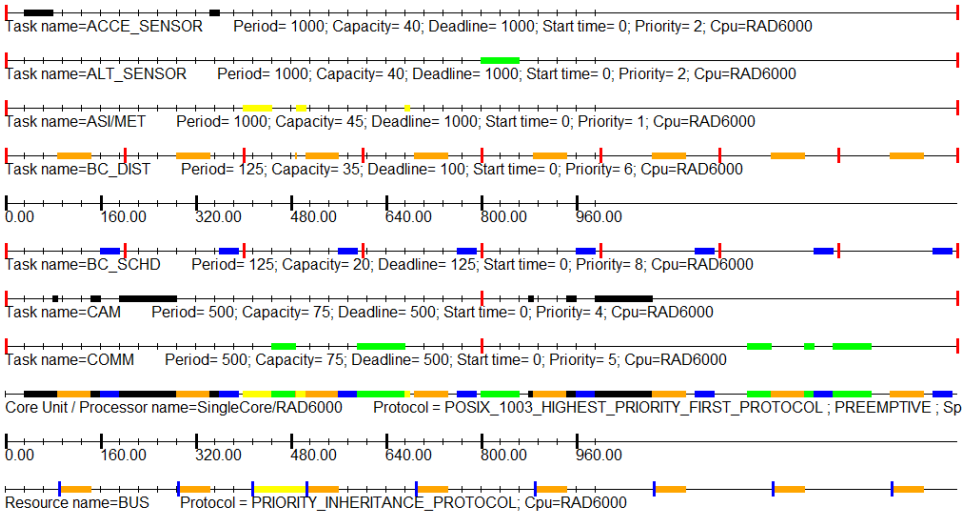

Fig. 4. Diagrama Temporal da simulação no Cheddar do escalonamento das tarefas no cenário 3 durante 1000 unidades de tempo.

Quando esta liberta a lock, a BC_DIST então interrompe o ASI/MET e prossegue com a sua execução completando a sua deadline.

Na figura 4 podemos observar a dinâmica deste protocolo. A tarefa COMM interrompe a ASI/MET (enquanto esta utiliza o recurso), de seguida a tarefa COMM é interrompida pela tarefa BC_DIST. No cenário anterior esta tarefa não conseguia ser concluída pois a tarefa ASI/MET possuía o lock do recurso, mas com o PIP quando a BC_DIST pede acesso ao recurso esta é interrompida, a tarefa ASI/MET herda a prioridade da BC_DIST e conclui a sua capacidade para, de seguida, BC_DIST continuar.

TABLE V
WORST TASK RESPONSE TIME CENÁRIO 3

Tarefa	WTRT
ACCE_SENSOR	225
ALT_SENSOR	540
BC_DIST	100
BC_SCHD	120
CAM	180
COMM	420
ASI/MET	425

D. Cenário 4

No cenário 3, foi referido que para resolver o problema do incumprimento da deadline foi utilizado o PIP (Priority Inheritance Protocol), este mecanismo permite a uma tarefa de menor prioridade herdar a prioridade de uma tarefa que fica bloqueada pelo mesmo recurso, no entanto, este mecanismo sozinho não prevém o chamado chain blocking e o deadlock pois, caso a tarefa necessite do recurso imediatamente após ao momento de ter sido chamada, não irá existir o mecanismo de herança e vamos ter um problema de deadlock.

Existe outros métodos de tetos de prioridades (CPP) que tentam resolver este problema e estão disponíveis na ferramenta Cheddar. Estes são o Original Priority Celling Protocol (OCPP) e o Immediate Priority Celling Protocol (ICPP). O primeiro protocolo define uma prioridade estática e uma prioridade dinâmica em cada tarefa, a prioridade estática é a prioridade definida por default e a prioridade dinâmica é dada pelo máximo entre a prioridade estática e qualquer uma das prioridades cujas tarefas estejam a ser bloqueadas pelo recurso. Existe depois uma prioridade para o recurso chamada de prioridade de teto, esta prioridade é definida pelo máximo de todas as prioridades das tarefas que partilham o recurso. Uma tarefa só consegue adquirir o recurso se a sua prioridade dinâmica for superior aos tetos de todos os recursos bloqueados por tarefas além da si mesma. O segundo protocolo funciona de forma parecida, no entanto a prioridade dinâmica de uma tarefa é definida pelo máximo entre a prioridade estática e o teto de qualquer um dos recursos bloqueados, desta forma uma tarefa não executa enquanto houver recursos que precisa a quererem ser usados.

Testámos ambos os protocolos no nosso sistema. O OCPP teve praticamente o mesmo comportamento que o PIP (mesmo número de mudanças de contexto, mesmo número de preempções e mesmos valores para o worst task response time), este resultado pode-se dar ao facto de estarmos a lidar apenas com um único recurso logo não existe mudanças de contexto que dificultam ou mostram a eficiência do OCPP. O ICPP mostrou-se mais vantajoso no nosso sistema, por ser um sistema com poucas tarefas e um recurso partilhado apenas, para além das tarefas terem uma capacidade consideravelmente reduzida, o bloqueio completo de tarefas que não necessitam do recurso para impedir o bloqueio deste fez diferença e diminuiu o worst task response time do BC_DIST de 100 para 90, como se pode ver na tabela VI, para além de diminuir o número de mudanças de contexto e o número de preempções.

Como os resultados do protocolo OCPP teve semelhante ao PIP, apresentamos apenas os resultados obtidos para a simulação do escalonamento com ICPP aplicado no BUS 1553 durante 1000 unidades de tempo.

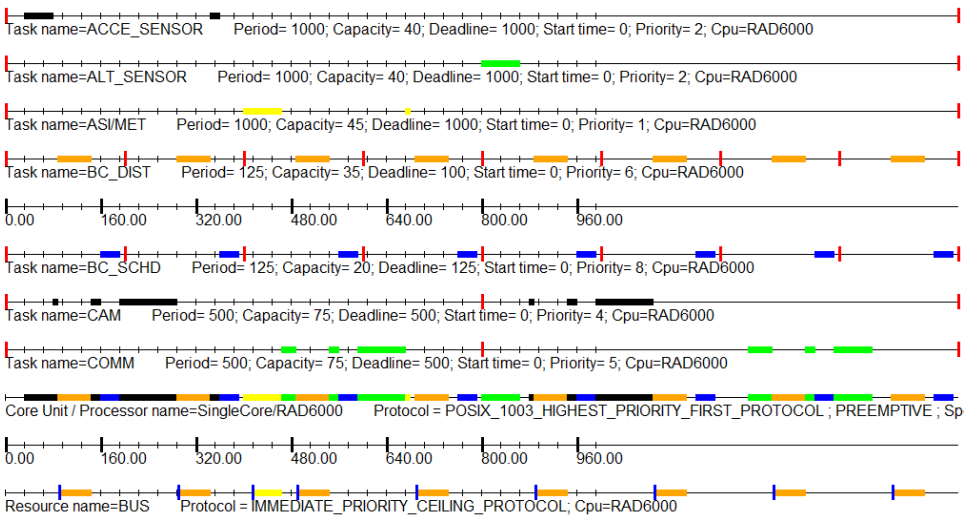

Fig. 5. Diagrama Temporal da simulação no Cheddar do escalonamento das tarefas utilizando ICPP durante 1000 unidades de tempo.

TABLE VI
WORST TASK RESPONSE TIME PROTOCOLO ICPP

Tarefa	WTRT
ACCE_SENSOR	225
ALT_SENSOR	540
BC_DIST	90
BC_SCHD	120
CAM	180
COMM	420
ASI/MET	425

Ao analisar o diagrama da figura 5 podemos ver que a dinâmica é diferente do cenário 3, sendo imediatamente notável que logo após à segunda deadline da task BC_DIST, esta não é executada durante uma unidade de tempo de forma a pedir a lock do recurso antes de ASI/MET voltar a ser executada até a libertar. O que acontece aqui é que a task ASI/MET só é interrompida pela task COMM após libertar a lock do recurso BUS, o que permite a task BC_DIST ser executa sem ser interrompida, pois não tem de pedir e esperar pela lock do recurso. Isto resulta, como tinha sido notado anteriormente, numa redução de preempções e mudanças de contexto tornando este escalonamento mais eficiente do que o do cenário anterior com o protocolo PIP. Caso o sistema operativo da sonda PathFinder tivesse este protocolo no seu core, acreditamos que esta seria uma melhor solução do que a implementada.

IV. DESAFIOS E DIFICULDADES

Nesta secção iremos referir as principais dificuldades sentidas no desenvolvimento do projeto.

Começando pela recriação do sistema da sonda PathFinder, o número de referências bibliográficas sobre os detalhes das tasks e escalonamento destas na sonda são mínimos. Este número reduzido de fontes tornou a recriação do sistema muito difícil, tendo havido vários componentes que ficamos na dúvida se deviamos de incluir (como por exemplo o VME BUS) e até nos parâmetros das tasks, tendo sido utilizado um pouco de trial and error em alguns do parâmetros (especialmente o offset de algumas tasks) de forma a recriar os cenários. Em relação à falta de fontes não se pode fazer muito, mas achamos que podiam ter sido dadas mais guidelines sobre as tasks extra de forma a não tornar esta parte do projeto tão ambígua.

Finalmente, em relação à ferramenta Cheddar, sentimos que a interface gráfica prova-se difícil de utilizar, especialmente na definição e edição das tarefas e na visualização do diagrama, existem poucas referências e materiais de estudo desta ferramenta, sendo a referência principal pouco útil na nossa opinião pois esta não explica muitos dos parâmetros na definição de tarefas e recursos, e portanto tivemos que ir testando várias definições de sistema até ficarmos satisfeitos com o resultado final. Para este caso achamos que talvez uma maior exposição a esta ferramenta nas aulas seria benéfica.

REFERENCES

[1] "What really happened on Mars", Glenn Reeves, The Risk Digest, Volume 19, Issue 54, January 1998.
[2] "What really happened on Mars Rover Pathfinder", Mike Jones, The Risk Digest, Volume 19, Issue 49, December 1997.
[3] "What the media couldn't tell you about the Mars Pathfinder", T. Durkin, Robot Science and Technology, 1998.
[4] Wikipedia. Mars Pathfinder. https://en.wikipedia.org/wiki/Mars_Pathfinder, 2023.