DESCRIÇÃO EM VHDL DE UMA TPU PARA CONTROLE DE MOTORES DE COMBUSTÃO INTERNA

MARIO H. CHAVES, HÉDIO TATIZAWA

Instituto de Energia e Ambiente, IEE, Universidade de São Paulo
CEP 05508-010, Avenida Professor Luciano Gualberto, 1289 – Cidade Universitária, SP, Brasil
E-mail: versão final

Abstract— This paper presents a VHDL library designed to facilitate the development of solutions regarding internal combustion engines control utilizing Field Programmable Gate Arrays (FPGA's). The focus of the library presented is the description in VHDL of a Time Processing Unit that has the function of timing and synchronism of actuators and the position of mechanical parts of an engine.

Keywords— FPGA, Time Processing Unit, Internal Combustion Engine.

Resumo— Este trabalho apresenta uma biblioteca descrita em VHDL criada para facilitar o desenvolvimento de soluções de controle para motores a combustão interna utilizando-se FPGA's. O foco da biblioteca é a descrição em VHDL de uma TPU que tem a função de temporização e sincronismo de atuadores com a posição das partes mecânicas do motor.

Palavras-chave— FPGA, TPU, Motores de Combustão Interna.

1 Introdução

A biblioteca apresentada nesse trabalho tem a função de servir como um ponto de partida para a aplicação de FPGA's (*Field Programmable Gate Array*) como TPU's (*Time Processing Unit*) em UCM's (Unidade de Controle do Motor) desenvolvidas para utilização em pesquisa e desenvolvimento de motores de combustão interna. Como observado por Guzella e Onder (2010), geralmente as UCM's apresentam uma peça de hardware chamada TPU, que é responsável por realizar o sincronismo entre os comandos de controle de atuadores e o movimento alternativo do motor. Nos trabalhos de Albaladejo (2013) e Pujatti (2007) são desenvolvidas UCM's visando a possibilidade do uso em pesquisa e desenvolvimento de sistemas de controle de motores de combustão interna. Com a biblioteca proposta, torna-se mais simples a implementação de um FPGA em sistemas como esses, permitindo o pesquisador explorar as características inerentes ao uso de FPGA's, como, processamento paralelo e alta velocidade de processamento.

De acordo com o sinal recebido dos sensores de posição do eixo do motor, o código da biblioteca proposta, estima a posição atual do eixo do motor e comanda bicos injetores, bobinas de ignição e saídas PWM de acordo com os comandos recebidos via SPI (*Serial Peripheral Interface*) por outro processador.

2 Estrutura Básica da Unidade de Controle

A biblioteca apresentada foi desenvolvida para operar em um sistema constituído de hardware encontrado em placas de prototipagem facilmente obtidas no mercado. A configuração e as funções dos processadores de dados foram feitos como descrito em Guzella e Onder (2010) (figura 1).

Para processamento de cálculos não relacionados ao processo de sincronismo, como interpolação de tabelas de períodos de comando de atuadores e a comunicação com periféricos, foi utilizado um microcontrolador ATSAM3X8E (Atmel Corp, 2015) encontrado na placa de prototipagem Arduíno Due.

Para cálculos de alta velocidade relacionados ao sincronismo de acionamentos com o sistema físico dinâmico é utilizado um FPGA Spartan 6 (Xilinx, 2011).

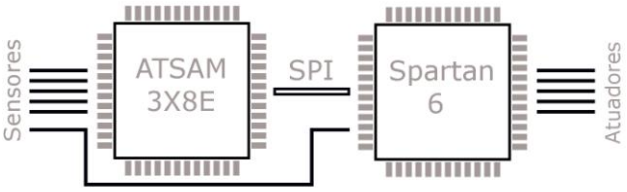

Figura 1. Configuração dos processadores

Depois de tratados, os sinais analógicos gerados pelos sensores da planta, são ligados aos conversores AD do microcontrolador. Os sinais digitais vindos dos circuitos condicionadores para sensores indutivos (sensores de posição do eixo do motor) são conectados diretamente ao FPGA, que verifica constantemente a posição do eixo e aciona os atuadores no momento desejado.

O sistema geral apresenta três tipos de entradas para sinais:

- Sinais com níveis de tensão variando entre 0 e 5V

- Entradas para sensores de temperatura resistivos (NTC ou PTC)
- Entradas digitais com circuito de tratamento para sensores de relutância variável (sensores de posição do eixo)

As saídas são todas digitais, sendo que quatro delas têm suporte a controle de corrente para comando de cargas indutivas.
O microcontrolador realiza, principalmente, os cálculos do algoritmo de controle que está sendo utilizado e envia ao FPGA os valores necessários para comando correto dos atuadores. Os valores enviados são os períodos de comando dos bicos injetores, das bobinas e do PWM (*Pulse Width Modulation*) e os ângulos de comando de cada atuador. Os ângulos de comando estão na faixa de 0 a 720° devido ao ciclo completo de um motor quatro tempos ser formado por duas voltas completas do eixo virabrequim (2 x 360°), o que compreende os ciclos de admissão, compressão, expansão e escape.
As funções de cálculos do microcontrolador não serão discutidas nesse trabalho, pois não são o objetivo principal e por variarem de acordo com a estratégia de controle a ser aplicada.
Utilizando as entradas digitais, o FPGA estima a atual posição do eixo. Com os valores de períodos de acionamento dos atuadores (recebidos via SPI – *Serial Peripheral Interface*) a posição de comando dos atuadores é calculada e comparada com a posição atual, caso a posição do eixo atual seja maior ou igual à posição calculada, a saída correspondente é acionada.

3 Núcleos em VHDL

Os processos de descrição dos circuitos em VHDL, síntese, simulação e geração dos arquivos binários a serem carregados no FPGA, foram realizados utilizando-se o *freeware ISE Design Suite* 2013, distribuído pela Xilinx.
Como sugerido em Mealy e Tappero (2016) toda a descrição dos núcleos em VHDL foi realizada utilizando-se somente as bibliotecas *IEEE Standard 1164* e *IEEE Numeric Std* para evitar problemas de compatibilidade entre bibliotecas e com o processamento do código. A seguir será apresentada a descrição básica dos núcleos desenvolvidos em VHDL que compõem a biblioteca.

3.1 Comunicação Serial

O núcleo de comunicação serial SPI escravo é responsável por receber e enviar dados de controle e monitoramento entre o microcontrolador e a TPU. Os dados recebidos via serial são dispostos em formato paralelo de 16 bits (figura 2). A descrição do núcleo foi realizada com base na descrição funcional da comunicação SPI encontrada em Atmel Corp, 2015.

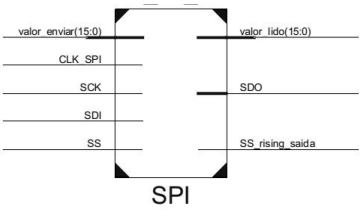

Figura 2. Núcleo de comunicação serial

A cada final de recebimento de dados o núcleo envia um comando com duração de um ciclo de clock pelo sinal "SS_rising_saida" que indica ao próximo núcleo o final de recebimento. No próximo ciclo de clock o dado está pronto para ser coletado por núcleos a diante no processo.

3.2 Separação e buffer de dados recebidos via comunicação serial

O núcleo de Separação e Buffer de dados é responsável por receber os dados do núcleo de comunicação serial e dispor em um buffer (figura 3). Os primeiros 6 bits de cada dado recebido indicam a posição dos outros 10 bits no buffer de saída.
O núcleo apresenta sete tipos de sinais de saída:

- BICO_PER_X e BOBINA_PER_X: Sinais de 30 bits que indicam o número de ciclos de clock do FPGA necessários para gerar o período desejado de comando dos bicos injetores e bobinas. Estes valores são calculados pelo algoritmo de controle executado no microcontrolador.
- BOBINA_PER_DESENERG: Sinal de 30 bits que indica o número de ciclos de clock do FPGA necessários para gerar o período mínimo de desenergização das bobinas de ignição. Considerou-se que esse valor poderia ser único para todas as bobinas devido ser usual utilizar-se bobinas de mesmo modelo.
- BICO_POS_X e BOBINA_POS_X: Sinais de 16 bits que indicam as posições do eixo em que os atuadores devem ser desligados. O sistema considera que o valor (2^{16}-1) equivale a 720° (ciclo completo para motores 4 tempos), portanto os valores dos sinais (0 a 2^{16}-1) equivalem a posições entre 0 e 720°.
- PWM_1_PER: Sinal de 30 bits que indica o período máximo de cada ciclo de comando de PWM.
- PWM_1_DC: Sinal de 10 bits que representa o *duty cycle* do PWM.

O processo de atualização dos sinais é realizado da seguinte maneira:

- Inicialmente os 10 bits de dados úteis são destinados à correta posição dentro de cada sinal no buffer.
- Depois que todos os bits de cada sinal são atualizados, o sinal "ATUADOR_ATUALIZADO" é carregado com um valor que indica qual sinal foi completamente atualizado.
- O sinal "SS_rising" é colocado em nível lógico '1' durante um ciclo de clock.

O comando do sinal "SS_rising" indica aos núcleos seguintes que o sinal atualizado deve ser lido.

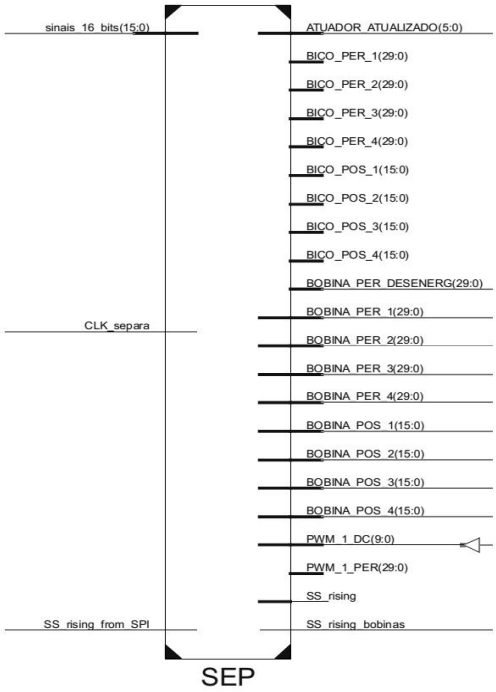

Figura 3. Núcleo de separação de dados e buffer

3.3 Estimação da posição do eixo do motor

A estimativa de posição atual do eixo do motor é realizada pelo núcleo de monitoração da posição do eixo do motor. Esse núcleo é responsável por monitorar os sensores de posição do eixo do motor e disponibilizar em um buffer a estimativa da posição do eixo.

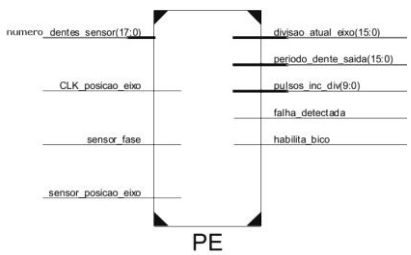

Figura 4. Núcleo de estimação da posição do eixo

Utiliza sinais do sensor de posição do eixo virabrequim e do sensor de fase, localizado no comando de válvulas, para estimar a posição atual do eixo.

O processo de estimação é realizado da seguinte maneira:

- Com o auxílio de um contador, o sistema obtém o número de ciclos de clock que ocorrem entre as bordas de subida do sinal vindo do sensor de posição do eixo, encontrando assim o período de duração de cada dente (considerando o uso de uma roda dentada).
- O valor encontrado de número de ciclos da leitura atual é comparado com a contagem da leitura anterior multiplicada por dois fatores que indicam os valores mínimo e máximo que a leitura atual pode ter (por exemplo, 0,25 e 1,25), caso a leitura atual esteja dentro dessa faixa, o sistema valida a nova leitura (Freescale, 2004).
- A nova leitura é disponibilizada para os núcleos de cálculos.
- A quantidade de ciclos medidos de um dente (cd) é divida pela quantidade teórica de divisões de posicionamento para um dente, ou seja, com qd sendo a quantidade de dentes da roda fônica, encontra-se com a equação 1 a quantidade teórica de ciclos de clock necessárias para que o eixo do motor se movimente aproximadamente 0,01098° (igual a $720°/2^{16}$).

$$pulsos\_inc\_div = \frac{cd * qd}{2^{16}} \qquad (1)$$

Paralelos aos cálculos de validação dos períodos dos dentes são realizados cálculos semelhantes para identificar se a nova leitura tem valor superior a 175% do valor da leitura anterior e inferior a 225% desse mesmo valor, caso a nova leitura esteja dentro dessa faixa, o núcleo considera que a falha de dente foi encontrada (considerando que seja falha de 1 dente) e uma *flag* interna é carregada com valor lógico '1'.

Caso a *flag* interna esteja carregada e o núcleo receba um sinal vindo do sensor de fase, o sistema entende que o eixo virabrequim realizou duas voltas completas e encerrou o ciclo, nesse momento o contador utilizado para indicar a divisão de posicionamento atual e a *flag* interna são zerados.

As faixas de tolerâncias utilizadas nas comparações são carregadas diretamente no código em VHDL do núcleo de estimação da posição do eixo, portanto, caso a quantidade de falhas de dentes mude, o código deve ser adaptado.

3.4 Cálculos para comando de bicos injetores e bobinas

O núcleo de cálculos é utilizado para encontrar a posição ideal do motor em que os atuadores devem ser acionados. Como padrão o sistema considera que a posição requerida em BOBINA_POS_X (BICO_POS_X) refere-se à posição em que as bobinas ou os bicos injetores seriam desativados, então o sistema calcula em que posição do eixo os atuadores devem ser ativados para que esses fiquem nesse estado durante o período indicado por BOBINA_PER_X (BICO_PER_X) e ao final desse período o eixo esteja na posição indicada por BOBINA_POS_X (BICO_POS_X).
As divisões de posicionamento do eixo em que os atuadores devem ser acionados são encontradas com o uso da equação 2.

$$divisão\_aciona\_x = POS\_X - d\_POS \qquad (2)$$

$$d\_POS = \frac{atuador\_PER\_X}{pulsos\_inc\_div} \qquad (3)$$

Sendo:

- divisão_aciona_x = Divisão de posicionamento em que o atuador deve ser ativado (divisão_aciona_bico ou divisão_aciona_bobina)
- POS_X = Divisão de posicionamento em que o atuador deve ser desativado (BOBINA_POS_X ou BICO_POS_X)
- atuador_PER_X = Período durante o qual o atuador deve permanecer ativo (BOBINA_PER_X ou BICO_PER_X)

Utiliza-se o mesmo núcleo para cálculos das posições de acionamentos para as bobinas de ignição e para os bicos injetores (figura 5), sendo que cada núcleo realiza cálculos para quatro atuadores.

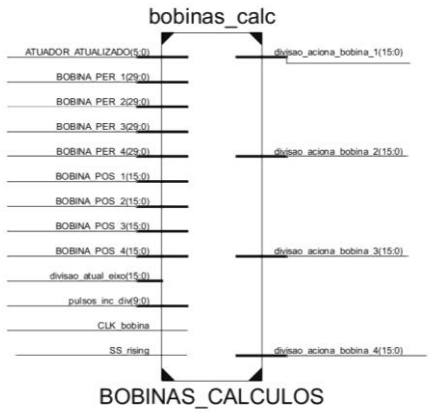

Figura 5. Núcleo de cálculos de posição de acionamento de atuadores

Na figura 6, observa-se um exemplo de comando da bobina de ignição para o cilindro 4, com avanço do ponto de ignição próximo a 0°. Os comandos de outros atuadores não são representados na figura 6 mas seriam similares, estando defasados em torno de 180° uns dos outros para um motor de quatro cilindros como considerado no exemplo.

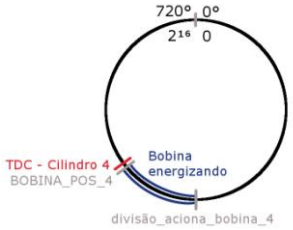

Figura 6. Exemplo de comando de bobina de ignição

3.5 Comando de bicos injetores e bobinas de ignição

Os núcleos de comandos dos bicos injetores e das bobinas de ignição (figura 7) operam quase que da mesma maneira, exceto que para o comando de bobinas de ignição é necessário garantir um período mínimo de desenergização.

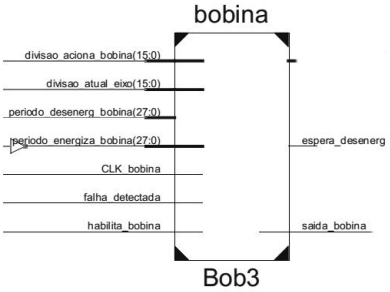

Figura 7. Núcleo de comando de atuadores.

Os núcleos verificam a cada pulso de clock se a divisão de posicionamento atual é maior que a posição calculada pelo núcleo de cálculos (sinais "divisão_aciona_bobina_x (bico)"), no caso dessa condição ser verdadeira, um contador é carregado com o valor do sinal BOBINAS_PER_X e BICOS_PER_X e inicia a contagem, nesse mesmo momento o atuador correspondente é ativado (bico ou bobina). Quando o contador é zerado os atuadores são desativados.
No caso das bobinas de ignição, no momento em que as bobinas são desativadas, um contador auxiliar carrega o valor de BOBINA_PER_DESENERG, carrega a saída "espera_desenerg" com nível lógico '1' e inicia a contagem.
Durante o período em que uma das saídas "espera_desenerg" dos *n* núcleos de comando de bobinas estiver acionada, uma lógica externa ao núcleo de comando desativa o acionamento das saídas dos outros núcleos.

Esse mecanismo é necessário, pois, nessa biblioteca não existe um núcleo de comando que aceita diversas posições de comando de atuadores, portanto, mesmo utilizando-se uma única bobina para *m* cilindros de um motor, ainda são necessários *m* núcleos de comando.

Quando o período de um ciclo do eixo do motor se torna menor que o período de energização da bobina, deixa de existir um intervalo entre energizações e uma falha de desenergização e consequente falha de ignição do combustível na câmara de combustão são observados, o que o mecanismo inibidor das saídas de ignição faz, é obrigar o sistema a sempre ter um período de desenergização da bobina.

3.6 PWM

O núcleo de PWM é responsável por gerar um sinal com largura de pulso variável (PWM). Realiza a divisão do valor do sinal "período_maximo" pelo valor do sinal "duty cycle".

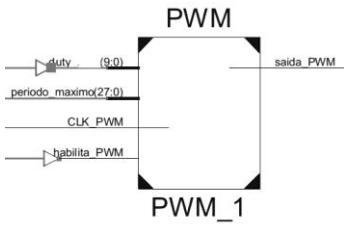

Figura 8. Núcleo PWM.

O valor de "período_maximo" é carregado em um contador que começa a ser decrementado, quando o valor do contador se iguala ao valor do sinal "duty cycle" a saída "saída_PWM" é colocada em nível lógico '1' até que o contador seja totalmente decrementado, momento em que a saída é colocada em nível lógico '0' e o contador é novamente carregado.

4 Conexão padrão dos Núcleos

A conexão padrão dos núcleos da biblioteca é apresentada na figura 9. Todos os núcleos respondem somente às variações do nível de clock do FPGA.

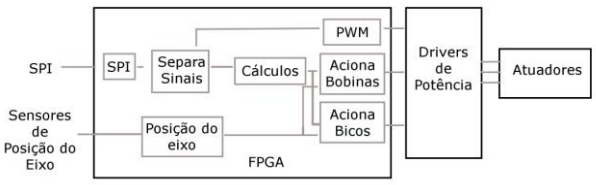

Figura 9. Conexão padrão dos núcleos

Somente uma série básica de blocos para um sistema padrão entrar em funcionamento foi utilizada no exemplo da figura 9. Como visto anteriormente cada núcleo de cálculos foi desenvolvido para quatro atuadores.

Em um sistema básico para um motor de quatro cilindros são necessários:

- 4 núcleos de comando de bobinas de ignição
- 4 núcleos de comando de bicos injetores
- 2 núcleos de cálculos de posição de acionamento dos atuadores
- 1 núcleo de comunicação SPI
- 1 núcleo de estimação da posição do eixo

No caso de comando de outros atuadores com o uso de PWM, basta declarar os núcleos de PWM e conecta-los como descrito anteriormente.

5 Resultados

A biblioteca descrita foi aplicada no hardware protótipo que pode ser observado na figura 10. Para testes o sistema foi utilizado unicamente para controle do acionamento das bobinas de ignição do motor.

A escolha de comando das bobinas de ignição para os testes do protótipo ocorre devido ao sincronismo de acionamento das bobinas ser mais crítico que o sincronismo de acionamento dos bicos injetores. Como os circuitos de comando das bobinas e dos bicos são quase idênticos entende-se que o correto funcionamento de um circuito implica no correto funcionamento do outro.

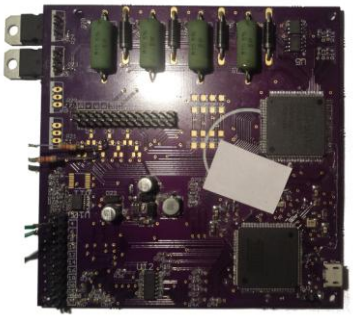

Figura 10. Placa protótipo (topo)

Outro fator para escolha do comando das bobinas de ignição durante os testes é a simplificação do código carregado no microcontrolador. A principal variável para controle do ponto de ignição é a velocidade angular do eixo do motor, com isso uma única tabela de interpolação (velocidade angular do eixo *x* ângulo de avanço de ignição) é o suficiente para testes na planta estudada.

Inicialmente foram realizados testes em bancada que indicaram uma precisão de comando dos atuadores de ao menos 1° (figura 11). A análise de precisão máxima foi limitada por características do instrumento utilizado (osciloscópio KEYSIGHT DSO1072). Na figura 11 observa-se um sinal simulado da roda

dentada, esse sinal foi injetado na entrada do FPGA para testes. Os períodos observados na figura 11 indicam os testes de comando realizados entre 90 e 90° (0, 90, 180, 270 e 360°) com a velocidade do eixo do motor simulada em 850 RPM.

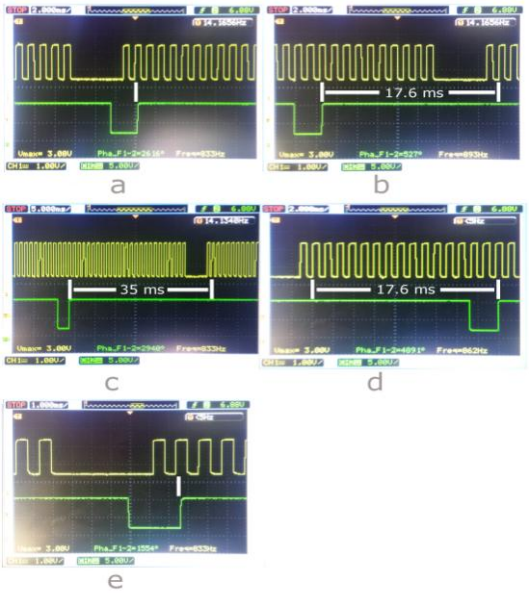

Figura 11. Medições em bancada de acionamento de atuadores entre 90 e 90° (17,6 e 17,6 ms).

Durante o funcionamento do motor, utilizando o sistema original de ignição, foi possível observar o comportamento do sistema experimental através de faixas brancas criadas na polia do comando de válvulas conforme um LED (*Light Emitting Diode*) de alta potência foi comandado pela saída de comando da bobina de ignição do primeiro cilindro (figura 12)

Figura 12. Ajuste de ponto de ignição do sistema experimental.

Utilizando-se um parâmetro de ajuste no microcontrolador, o comando dos cilindros foi corrigido para que o motor em marcha lenta tivesse 14° de avanço de ignição. Com o ponto de ignição ajustado em marcha lenta e com a sequência correta de acionamento das saídas, o motor foi colocado em funcionamento com o sistema experimental.

6 Conclusão

Apesar do sistema apresentado ainda estar em desenvolvimento, observa-se o sucesso no desenvolvimento de uma biblioteca funcional para sincronismo de atuadores com o sistema mecânico estudado, sendo que tal sistema pode ser adaptado para comando de qualquer tipo de atuadores que trabalham sincronizados com algum mecanismo que tem movimentos cíclicos. Pretende-se em trabalhos futuros melhorar o sistema de comunicação e separação de dados vindos pela porta serial devido aos núcleos atuais serem muito limitados.

Agradecimentos

Agradeço ao professor Hédio Tatizawa pelo apoio, orientação e conhecimentos passados durante esse trabalho, ao Instituto de Energia e Ambiente pelo ambiente de trabalho e ao CNPQ pelo auxílio financeiro.

Referências Bibliográficas

Albaladejo, F. S. Desenvolvimento de uma Unidade de Gerenciamento Eletrônico para Motores de Combustão Interna do Ciclo Otto. Dissertação de Mestrado. Programa de Pós-Graduação em Microeletrônica da Escola Politécnica da Universidade de São Paulo, São Paulo, 2013.

Atmel Corp. (2015). SAM3X / SAM3A Series - Atmel | SMART ARM-based MCU. Available at: http://www.atmel.com/pt/br/Images/Atmel-11057-32-bit-Cortex-M3-Microcontroller-SAM3X-SAM3A_Datasheet.pdf [Accessed 13 Aug. 2015].

Freescale (2004). TPU Time Processor Unit Reference Manual (Including the TPU2). [ebook] pp.A15. Available at: http://cache.freescale.com/files/microcontrollers/doc/ref_manual/TPURM.pdf [Accessed 21 Mar. 2016].

Guzella, L. and Onder, C.H. (2010). Introduction to Modeling and Control of Internal Combustion Engine Systems. Second Edition, Springer-Verlag Berlin Heidelberg, Germany.

Mealy, B. and Tappero, F. (2016). *Free Range VHDL*. 1st ed. [ebook] Available at: http://freerangefactory.org/pdf/free_range_vhdl.pdf [Accessed 21 Mar. 2016].

Pujatti, F. J. P. Desenvolvimento de um Sistema de Gerenciamento Eletrônico para Motores de Ignição por Centelha. Tese de Doutorado. Programa de Pós-Graduação em Engenharia Mecânica da Universidade de Minas Gerais, Minas Gerais, 2007.

Xilinx Inc. (2011). Spartan 6 - Family Overview. Available at: http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf [Accessed 21 Mar. 2016].