



IMPLEMENTAÇÃO DE UM CONTROLADOR FUZZY PARA CONVERSOR BUCK EM MICROCONTROLADOR AVR DE 8 BITS

Eduardo da Veiga Beltrame¹, Clóvis Antônio Petry², Samir Ahmad Mussa³

Resumo: Este trabalho propõe o estudo e desenvolvimento de um controlador fuzzy para conversores estáticos. Assim, é apresentada a implementação de um controlador fuzzy para conversor buck em microcontrolador AVR de 8 bits. Com a utilização do conversor analógico-digital (A/D) e gerador PWM (modulação por largura de pulso) embutidos no microcontrolador, a implementação foi possível com poucos componentes externos. Ao invés de executarem-se os cálculos em tempo real, foram utilizadas tabelas de consulta para implementação do controlador, obtendo-se respostas estáveis para o conversor buck. Ao final são apresentados os resultados experimentais do sistema implementado, constatando-se o funcionamento correto e a boa performance do controlador utilizado.

Palavras-chave: Lógica fuzzy. Controle digital. Microcontroladores. Conversores CC-CC.

Abstract: This document presents the implementation of a fuzzy controller for the buck converter using a 8 bit AVR microcontroller. Using the integrated analogic-to-digital (A/D) and PWM (pulse width modulation) generator, the implementation was made possible with few external components. Instead of performing the calculations in real time, look-up tables were used for the controller implementation, and stable responses were obtained for the buck converter. Experimental results for the implemented prototype are presented, showing the functioning of the system and the controller performance.

Keywords: Lógica fuzzy. Digital control. Microcontrollers. CC-CC converters.

¹ Acadêmico do curso técnico integrado de eletrônica do DAELN do IFSC. <veigabeltrame@gmail.com>.

² Professor do DAELN, campus Florianópolis, do IFSC <petry@ifsc.edu.br>.

³ Professor do Departamento de Engenharia Elétrica e INEP da UFSC <samir@inep.ufsc.br>.

1. INTRODUÇÃO

As técnicas de controle clássico têm sido aplicadas com sucesso a uma miríade de situações. A implementação de controladores não-lineares, entretanto, tem sido imensamente facilitada com o crescimento exponencial da capacidade de processamento de circuitos digitais.

Lógica fuzzy (ZADEH, 1965) é uma das opções para elaboração de um controlador, fugindo das metodologias de projeto tradicionais. A utilização de regras e variáveis linguísticas é o caminho para obtenção de um sistema de inferência fuzzy (SIMÕES; SHAW, 2007; MENDEL, 1995), o qual pode ser traduzido como o mapeamento de uma superfície n-dimensional (dependendo no número de variáveis de entrada e saída), ou seja, tabelas.

Como todas as possíveis saídas de um sistema fuzzy podem ser armazenadas numa tabela, em vez

de se realizarem os cálculos em tempo real, para a implementação do controlador foram utilizadas tabelas de consultas gravadas no microcontrolador.

Para obter as entradas do algoritmo fuzzy, o microcontrolador amostra a saída a intervalos regulares, utilizando o conversor A/D embutido. O valor do erro (e) é obtido comparando-se uma referência interna (V_{ref}) com a saída do conversor A/D (V_o), de acordo com a Equação 1. Já a variação do erro (ce) é obtida pela diferença entre o erro atual e o erro da amostragem anterior (e_{old}), conforme a Equação 2.

$$e = V_{ref} - V_o \quad [1]$$

$$ce = e - e_{old} \quad [2]$$

O algoritmo fuzzy de controle é composto por três estágios, descritos em detalhes na próxima seção. São eles:

- fuzzificação, onde os valores de entrada amostrados são convertidos em variáveis linguísticas, ou conjuntos *fuzzy*, adequados a serem tratados por lógica *fuzzy*;
- inferência ou tomada de decisões, onde são aplicadas as regras definidas para o algoritmo; e defuzzificação, que é a tradução do resultado *fuzzy* do processo de inferência num único valor não *fuzzy*, o valor da mudança no ciclo ativo (δd), que será somado ao ciclo ativo anterior (d_{old}), obtendo-se o novo ciclo ativo (d), conforme a Equação 3.

$$d = d_{old} + \delta d \quad [3]$$

Essa ação tem um efeito integrador, diminuindo o erro em regime permanente. O ciclo ativo assim calculado é aplicado pelo gerador PWM do microcontrolador no conversor buck.

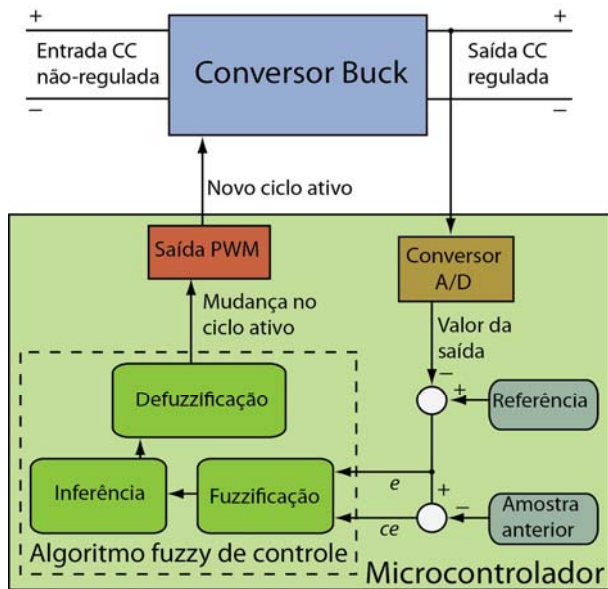


FIGURA 1 – Diagrama de blocos do projeto.

2. O ALGORITMO FUZZY DE CONTROLE

O algoritmo *fuzzy* de controle foi elaborado com a ajuda da *Fuzzy Logic Toolbox* (FUZZY, 2002) do *software* MATLAB, o qual também permitiu a simulação do algoritmo junto com a planta do conversor em seu poderoso ambiente de simulação, *Simulink*.

O aspecto geral do algoritmo foi baseado na literatura existente (GUPTA *et al.*, 1997; OFOLI, RUBAAI, 2005), e suas etapas de funcionamento são descritas a seguir.

2.1. Funções de pertinência de entrada

O primeiro passo de um sistema *fuzzy* é a atribuição de pertinências às entradas. Pertinência, denotada por μ , é o grau com que atribuímos uma

qualidade a uma variável de entrada. Para o nosso algoritmo, tanto o erro e quanto a variação do erro ce possuem as mesmas qualidades, ou conjuntos *fuzzy*:

NB	Negative Big (negativo grande)
NM	Negative Medium (negativo médio)
NS	Negative Small (negativo pequeno)
ZE	Zero
PS	Positive Small (positivo pequeno)
PM	Positive Medium (positivo médio)
PB	Positive Big (positivo grande)

Cada qualidade é definida por uma função de pertinência, que atribui para o valor de entrada uma pertinência a qualidade associada. Como temos 7 qualidades, temos 7 funções de pertinência, mostradas na Figura 2. Utilizaram-se apenas funções triangulares e trapezoidais, mas elas podem ter qualquer formato arbitrário. Geralmente a pertinência é normalizada para ficar entre 0 (integralmente falso) e 1 (integralmente verdadeiro), mas como estamos lidando com valores de 8 bits vindos do conversor A/D, essa faixa constitui-se dos valores inteiros de 0 a 255. O mesmo vale para as entradas.

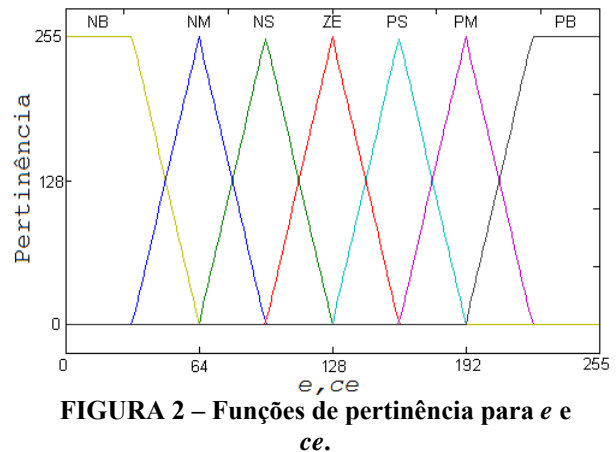


FIGURA 2 – Funções de pertinência para e e ce .

Note-se também na Figura 2 que os valores de entrada estão deslocados em 128 para a direita, sendo 128 o centro da escala (ZE). Assim sendo, um erro de 128 significa um erro nulo. Se, por exemplo, tivermos e de 182 e ce de 132, e pertencerá aos conjuntos PS e PM com uma pertinência $\mu_{PS}(e)$ de 79 e $\mu_{PM}(e)$ de 176 e ce aos conjuntos ZE e PS com pertinências de $\mu_{ZE}(ce)$ com 223 e $\mu_{PS}(ce)$ com 32. Para todas as outras funções, a pertinência de e e ce será nula.

2.2. Funções de saída

Assim como temos qualidades de entrada, temos qualidades de saída, que precisam ser definidas, cujas pertinências serão dadas pelas regras, descritas a seguir. Para as funções de saída do algoritmo foram definidas 9 qualidades:

NVB	Negative Very Big (negativo muito grande)
NB	Negative Big (negativo grande)
NM	Negative Medium (negativo médio)
NS	Negative Small (negativo pequeno)
ZE	Zero
PS	Positive Small (positivo pequeno)
PM	Positive Medium (positivo médio)
PB	Positive Big (positivo grande)
PVB	Positive Very Big (positivo muito grande)

Suas funções são apresentadas na Figura 3. Utilizaram-se apenas *fuzzy singletons*, ou seja, funções cujo peso concentra-se num único ponto.

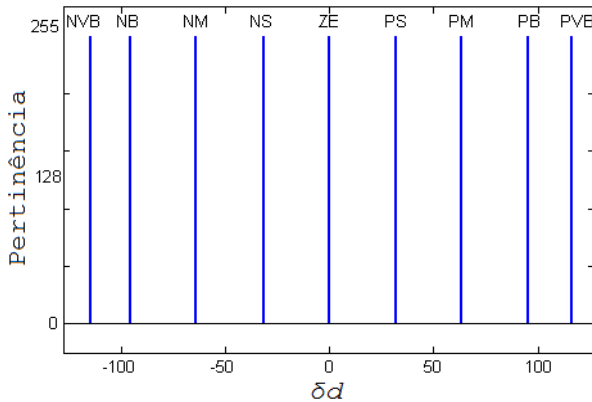


FIGURA 3 – Funções de saída do controlador.

2.3. Regras

Regras são a maneira pela qual as pertinências da entrada afetam as pertinências da saída. No nosso controlador *fuzzy*, cada regra associa uma combinação das pertinências de entrada com uma saída, tomando, por exemplo, a forma:

“Se e é **NS** e ce é **ZE** então δd será **NS**”.

Como temos duas entradas com 7 qualidades cada, para cobrir todas as possibilidades de entradas precisamos de 49 regras. A Tabela 1 apresenta as regras de controle.

TABELA 1 – Regras de controle.

		Variação do erro (ce)						
		NB	NM	NS	ZE	PS	PM	PB
Erro (e)	NB	NVB	NVB	NVB	NB	NM	NS	ZE
	NM	NVB	NVB	NB	NM	NS	ZE	PS
	NS	NVB	NB	NM	NS	ZE	PS	PM
	ZE	NB	NM	NS	ZE	PS	PM	PB
	PS	NM	NS	ZE	PS	PM	PB	PVB
	PM	NS	ZE	PS	PM	PB	PVB	PVB
	PB	ZE	PS	PM	PB	PVB	PVB	PVB

A pertinência da saída de cada regra é inferida como sendo a menor das duas pertinências de entrada. Assim, continuando com nosso exemplo da sessão 2.2, teríamos 4 combinações de e e ce que ativariam regras: (**PS**, **ZE**), (**PS**, **PS**), (**PM**, **ZE**), e

(**PM**, **PS**). A saída inferida (z) de cada regra pode ser descrita pela Equação 4.

$$\mu(z) = \min(\mu(e), \mu(ce)) \quad [4]$$

Teremos então a pertinência dos conjuntos de saída indicados nas Equações 5 a 8.

$$\mu_{PS}(z) = \min(\mu_{PS}(e), \mu_{PS}(ce))$$

$$\mu_{PS}(z) = \min(79, 32)$$

$$\mu_{PS}(z) = 32 \quad [5]$$

$$\mu_{PM}(z) = \min(\mu_{PM}(e), \mu_{ZE}(ce))$$

$$\mu_{PM}(z) = \min(176, 223)$$

$$\mu_{PM}(z) = 176 \quad [6]$$

$$\mu_{PB}(z) = \min(\mu_{PM}(e), \mu_{PS}(ce))$$

$$\mu_{PB}(z) = \min(176, 32)$$

$$\mu_{PB}(z) = 32 \quad [7]$$

$$\mu_{PS}(z) = \min(\mu_{PS}(e), \mu_{ZE}(ce))$$

$$\mu_{PS}(z) = \min(79, 223)$$

$$\mu_{PS}(z) = 79 \quad [8]$$

2.4. Defuzzyficação

Para defuzzyficar o resultado inferido pelas regras, é necessário primeiro atribuir a cada conjunto de saída uma única pertinência. Aqui isso é feito escolhendo-se a maior das pertinências associadas a ele. Uma vez que cada conjunto tenha apenas uma pertinência, eles são agregados num único conjunto, como na Figura 4. Defuzzyficar a saída consiste em extrair um único valor numérico a partir deste conjunto agregado. Utilizamos o método do cálculo do centróide, onde encontramos o ponto em que se localiza o centro de gravidade.

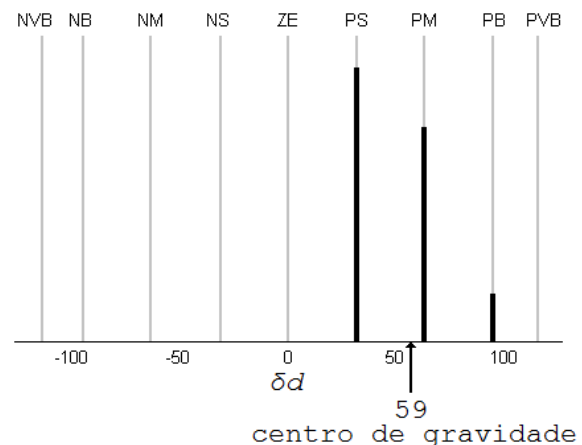


FIGURA 4 – Conjunto agregado e cálculo do centróide.

Quando estamos trabalhando com valores discretos, o cálculo do centróide consiste em realizar a soma ponderada de todas as massas, isto é, a soma de todas as massas multiplicadas pela sua localização(p), indicada na Equação 9

$$\delta d = \frac{\sum \mu p}{\sum \mu} \quad [9]$$

Finalizando nosso exemplo e realizando o cálculo do centróide, encontramos que o valor a ser somado ao ciclo ativo, que vai de 0 a 255, é 59

2.5. Superfície de controle

A Figura 6 apresenta a superfície de controle mostrando todas as possíveis entradas e saídas do controlador.

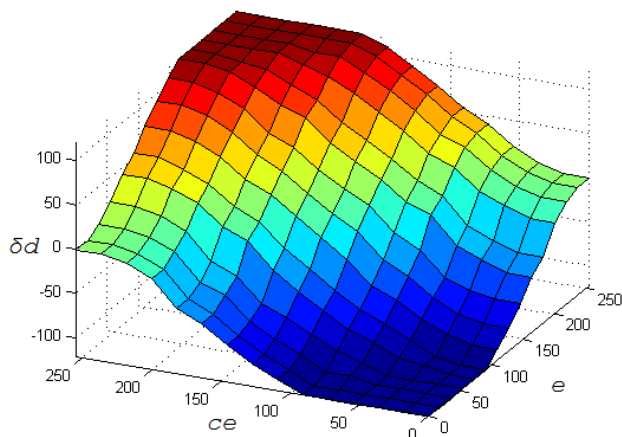


FIGURA 5 – Superfície de controle.

3. IMPLEMENTAÇÃO

O algoritmo descrito na seção anterior foi traduzido para linguagem C e simulado com o

software PSIM controlando um conversor buck. Após sua validação, o código foi usado para gerar as tabelas de consulta que foram implementadas no microcontrolador.

3.1. O microcontrolador ATmega328P

Para implementação, utilizou-se o microcontrolador AVR ATmega328P, fabricado pela Atmel. Com a utilização de um cristal externo, ele opera em até 20 MHz. Possui 32 kB de memória de programa, 2 kB de memória de acesso aleatório (RAM), três temporizadores/contadores, seis canais PWM e seis canais para o conversor A/D.

3.2. Questões encontradas

Em vez de se realizarem os cálculos para encontrar o novo ciclo ativo a cada amostragem, optou-se pela utilização de tabelas de consulta (HE; NELMS, 2005) reduzindo o processamento necessário ao custo de um dispêndio de memória de programa. Como o erro e a variação do erro são variáveis de 8 bits, que podem assumir 256 valores, e tendo cada saída o tamanho de um byte, a tabela de consulta completa teria de ter 64 kB.

Por causa da memória limitada do microcontrolador, foram geradas duas tabelas de 64x64 bytes: uma de baixa resolução, para regiões periféricas de operação, por onde o controlador só passaria durante transitórios, e uma de alta resolução para a região central da tabela, em torno da qual o controlador opera em regime permanente.

A Tabela 3, de 16x16, é uma versão em baixa resolução das tabelas implementadas. Note-se que os valores da tabela refletem as alturas da superfície de controle.

TABELA 3 – Tabela de consulta do controlador fuzzy (baixa resolução).

		Variação do erro (ce)															
		240	224	208	192	176	160	144	128	112	96	80	64	48	32	16	0
Erro (e)	240	127	127	127	127	127	127	126	111	95	79	63	47	31	15	0	0
	224	127	127	127	127	127	127	126	111	95	79	63	47	31	15	0	0
	208	127	127	127	127	126	126	126	111	95	79	63	47	31	15	0	0
	192	127	127	127	127	126	119	126	111	95	79	63	47	31	15	0	0
	176	127	127	126	126	124	109	124	109	93	77	61	45	29	13	-1	-1
	160	127	127	126	119	109	95	109	95	77	63	45	31	13	-1	-17	-17
	144	127	127	124	109	93	77	93	77	62	47	30	15	-3	-19	-33	-33
	128	111	111	109	95	77	63	77	63	47	48	46	15	-19	-33	-49	-49
	112	96	96	93	78	61	45	61	45	30	46	56	10	-35	-51	-65	-65
	96	96	96	91	63	45	31	45	31	15	15	10	-17	-51	-65	-81	-81
	80	32	32	21	30	29	13	29	13	-3	-19	-35	-51	-67	-83	-97	-97
	64	32	32	30	20	13	-1	13	-1	-19	-33	-51	-65	-83	-97	-113	-113
	48	31	31	29	13	-3	-19	-3	-19	-35	-51	-67	-83	-98	-114	-128	-128
	32	15	15	13	-1	-19	-33	-19	-33	-51	-65	-83	-97	-114	-120	-128	-128
	16	0	0	-1	-17	-33	-49	-35	-51	-67	-83	-98	-114	-128	-128	-128	-128
	0	0	0	-1	-17	-33	-49	-51	-65	-83	-97	-114	-120	-128	-128	-128	-128

O microcontrolador opera a 20 MHz, para gerar as interrupções para uma nova amostragem utilizou-se o estouro de um temporizador, de forma que ocorrem com uma frequência de aproximadamente 9,8 kHz. O PWM, com uma resolução de 8 bits, opera a 78,125 kHz. As amostragens ocorrem então a cada 8 ciclos de chaveamento.

3.3. Montagem

Para os testes foi projetado e montado um conversor *buck*, dimensionado para uma entrada de 24 V e saída de 12 V com corrente de 1 A. O esquemático do circuito utilizado para os testes é apresentado na Figura 6.

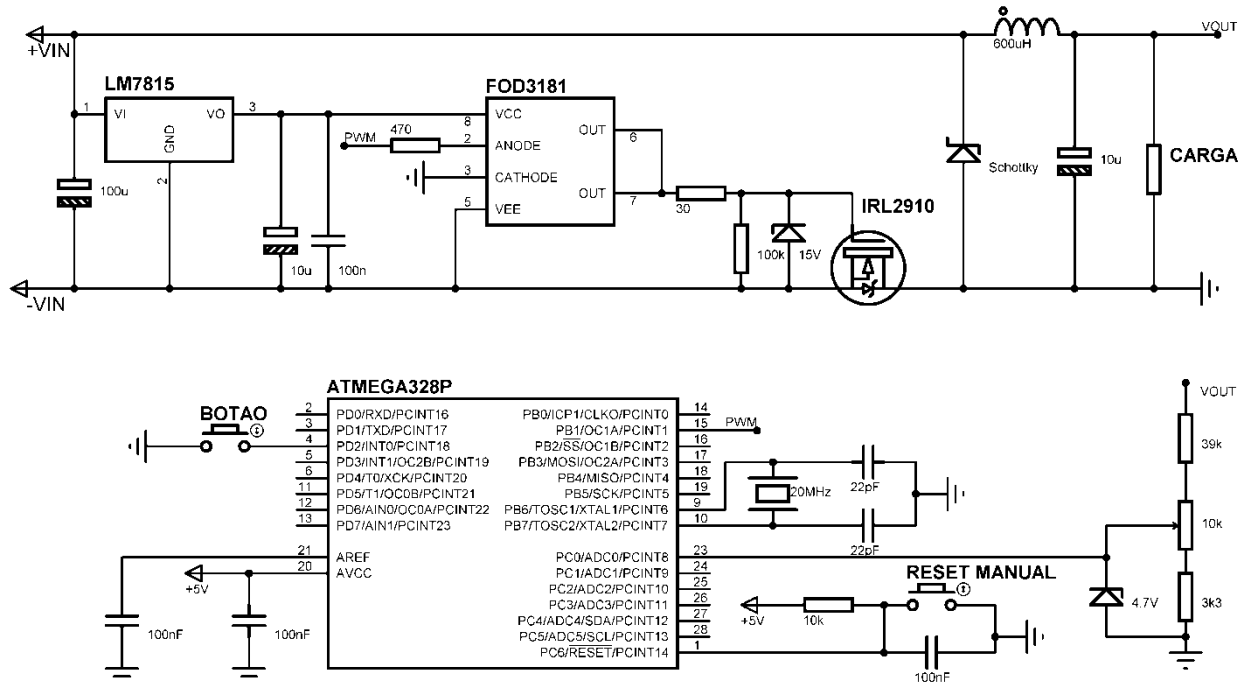


FIGURA 6 – Esquemático da montagem realizada.

Para os testes, o conversor foi alimentado com 24 V com uma carga de 12 Ω ou 48 Ω . O microcontrolador era reiniciado e, quando fosse apertado um botão, passava a realizar a leitura da saída e ajuste do ciclo ativo. Para ajustar a tensão de saída utilizou-se um potenciômetro no divisor de tensão ligado ao pino do conversor A/D. Os testes de variação de carga foram feitos com uma chave, ligando e desligando resistores em paralelo com a carga.

4. RESULTADOS EXPERIMENTAIS

As formas de onda de saída foram registradas com um osciloscópio digital da Tektronix. As respostas ao degrau e transitório de variação de carga são apresentadas nas Figuras 7, 8(a) e 8(b).

Os transitórios de partida duram cerca de 1,2ms, não ocorrendo *overshoot* mesmo com variação da carga de partida.

O transiente de carga durou cerca de 2,6 ms. Durante os testes realizados o controlador conseguiu eliminar o erro em regime permanente.

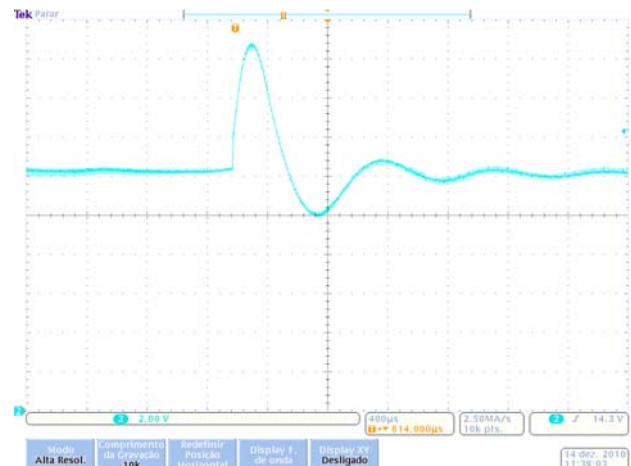


FIGURA 7 – Transitório de 48 Ω para 12 Ω . Escalas de 2V/div e 400 μ s/div.

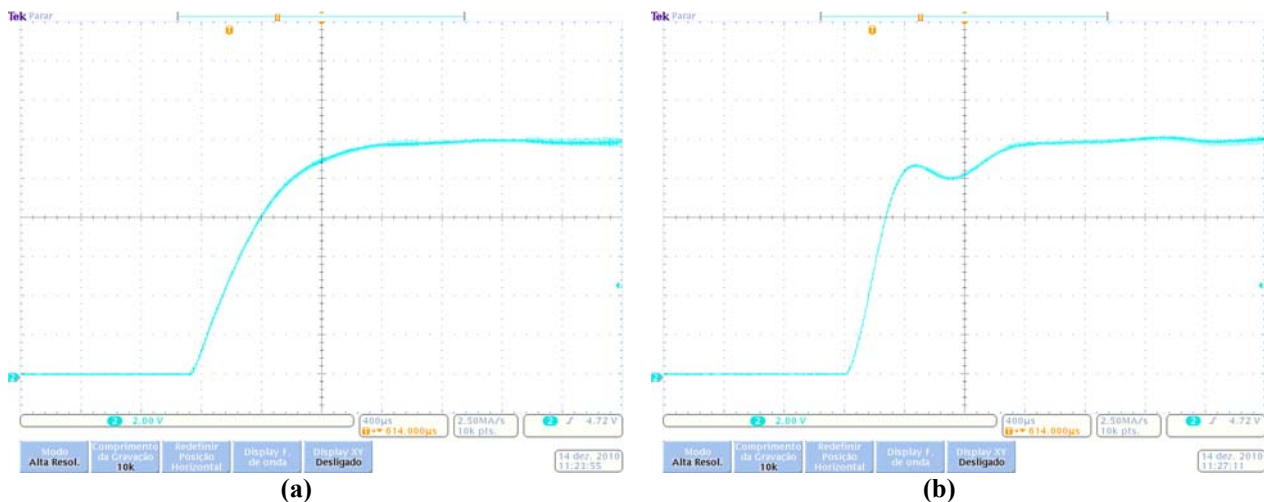


FIGURA 8 – Resposta ao degrau com cargas de (a) 12 Ω e (b) 48 Ω . Escalas de 2V/div e 400μs/div.

5. CONCLUSÃO

Foi descrito um algoritmo de controle *fuzzy* para conversor *buck*, implementado num microcontrolador de 8 bits. A utilização de tabelas de consulta, graças à memória disponível no microcontrolador, permitiu evitar as complexidades de se implementar um algoritmo *fuzzy* em tempo real. Os periféricos disponibilizados pelo microcontrolador, conversor A/D e gerador PWM simplificaram a montagem, permitindo a implementação do controlador com poucos componentes externos.

O controlador apresentou resultados satisfatórios, conseguindo controlar a tensão de saída sem erro mesmo com variações na carga.

Em retrospecto, um controlador *fuzzy* apresenta várias alternativas em relação aos controladores convencionais, especialmente durante o projeto do controlador, mas sua utilização só se torna interessante quando um controlador clássico não pode ser usado, ou seu desempenho deixa a desejar.

Agradecimentos

Um agradecimento aos professores do departamento de eletrônica do IFSC.

REFERÊNCIAS

- FUZZY Logic Toolbox User's Guide, The MathWorks, 2002.
- GUPTA, T. R. R. *et al.* Implementation of a Fuzzy Controller for DC-DC Converters Using an Inexpensive 8-bit Microcontroller. **IEEE Transactions on Industrial Electronics**, vol. 44, 1997.
- HE, D. ; NELMS, R.M. **Fuzzy Logic Peak Current-Mode Control for DC-DC Converters Using an Inexpensive 8 Bit Microcontroller**, IEEE, 2005.
- MENDEL, J.M. **Fuzzy Logic Systems for Engineering: A Tutorial**, IEEE, 1995
- OFOLI, A. R.; RUBAAI, A. **Real-Time Implementation of a Fuzzy Logic Controller for Switch-Mode Power-Stage DC-DC Converters**, IEEE, 2006.
- SIMÕES, M. G.; SHAW, I. S. **Controle e Modelagem Fuzzy**, Editora Blucher, São Paulo, 2007.
- ZADEH, L.A. Fuzzy sets. **Information and Control**, vol. 8, 1965.