Implementação e Análise de Desempenho de um Protótipo de Sistema OFDM de Baixa Resolução com Hardware de Baixo Custo

Eder de Souza, Demerson N. Gonçalves e João Dias

***Resumo*— Com o objetivo de analisar a viabilidade de implementação de elementos de novos sistemas de comunicação sob uma abordagem SDR, este trabalho apresenta a prototipagem de um enlace de comunicação OFDM de baixa resolução usando a ferramenta de código aberto GNU Radio e os dispositivos de baixo custo HackRF One e RTL-SDR Blog V3. Os resultados se mostraram promissores, permitindo a concepção e teste de sistemas de comunicação digital de uma forma prática e levando em consideração a real condição do canal e deficiências do *front-end*.**

***Palavras-Chave*— Prototipagem, Q-OFDM, SDR.**

***Abstract*— In order to analyze the feasibility of implementing elements of new communication systems under an SDR approach, this work presents the prototyping of a low resolution OFDM communication link using the open source tool GNU Radio and the low cost devices HackRF One and RTL-SDR Blog V3. The results were promising, allowing the design and testing of digital communication systems in a practical way and taking into account the real condition of the channel and deficiencies of the front-end.**

***Keywords*— Prototyping, Q-OFDM, SDR.**

I. INTRODUÇÃO

Novas aplicações de baixa latência, grande densidade de conexão e conectividade onipresente requerem um sistema de comunicação eficiente e flexível, uma vez que as características das camadas físicas (PHY) e de acesso ao meio (MAC) afetam o desempenho geral do sistema em termos de eficiência energética, eficiência espectral, *throughput* alcançável, qualidade de serviço (QoS), etc [1]. Por outro lado, a prototipagem de um sistema de comunicação precisa ser flexível de modo que possa ser capaz de dar suporte a novos algoritmos e testar diferentes formas de uso. O rádio definido por software (SDR) é uma abordagem atraente para prototipagem flexível de sistemas de comunicação onde a funcionalidade dos sistemas depende do software implementado em um dispositivo programável [2].

Em muitos trabalhos tem-se visto o uso do USRP (Universal Software Radio Peripheral) para prototipagem de sistemas de comunicação [3]. Porém, o modelo mais barato desse SDR custa em torno de dois mil dólares a unidade [4], enquanto o custo total (SDRs para transmissão e recepção, mais as antenas) gasto neste trabalho é de menos de duzentos dólares.

Eder de Souza, Engenharia Eletrônica, CEFET/RJ, Rio de Janeiro - RJ, e-mail: ederoliveira@id.uff.br; Demerson N. Gonçalves, Coordenação de Licenciatura em Matemática, CEFET/RJ, Petrópolis - RJ, e-mail:demerson.goncalves@cefet-rj.br; João Dias, Departamento de Engenharia de Telecomunicações, CEFET/RJ, Rio de Janeiro - RJ, e-mail: joao.dias@cefet-rj.br.

O trabalho [5] é um estudo do uso de um equalizador LRA-MMSE em um sistema OQ-OFDM com baixa resolução. O trabalho [6] inicia uma implementação básica de um sistema OFDM com o uso de SDRs de baixo custo. Nesse sentido, este trabalho representa um avanço na investigação da performance de um conversor A/D de baixa resolução através de prototipagem. O objetivo, portanto, é apresentar a implementação de um enlace de comunicação usando o processador de um laptop padrão para as funções de banda base digital e o HackRF One e RTL-SDR como hardware de interface RF. O sistema usa a técnica de transmissão por divisão de frequências ortogonais (OFDM) com baixa resolução no quantizador do lado da recepção. Como camada de software, foi usado o GNU Radio, um conjunto de bibliotecas de código aberto escrito em C++ e Python.

O restante do trabalho está dividido da seguinte forma: o modelo do sistema de comunicação OFDM é apresentado na Seção II; na Seção III é apresentado o desenvolvimento deste modelo e uma breve descrição do quantizador implementado; na Seção IV são apresentados alguns resultados de desempenho da prototipagem realizada; na Seção V são feitas as conclusões do trabalho.

II. OFDM

OFDM é um esquema de modulação multiportadora baseada na divisão do fluxo serial de dados em grupos paralelos de fluxos que são transmitidos em subportadoras ortogonais. Quando um sinal OFDM atravessa um canal dispersivo no tempo ou seletivo em frequência, as subportadoras são afetadas apenas por um canal constante ou plano na frequência. Nesse tipo de canal, o sinal pode ser distorcido por interferência intersimbólica (ISI). Este problema é resolvido no OFDM inserindo uma cópia da última parte do símbolo, também conhecida como prefixo cíclico (CP), no início, para absorver os espalhamentos de atraso de canal. O sinal OFDM pode ser expresso no domínio do tempo por [7]

$$\mathbf{x}[n] = \sum_{k=0}^{K-1} \mathbf{s}_k \, e^{j2\pi \frac{k}{K} n}, \tag{1}$$

onde $\mathbf{s}_k$ é o símbolo de dados na $k-ésima$ subportadora e K é o número de subportadoras no símbolo OFDM. O sinal na entrada do receptor pode ser escrito por $\mathbf{y} = \mathbf{x} * \mathbf{h} + \omega$, onde $\mathbf{y} \in \mathbb{C}^{(K+CP+N_p-1)\times 1}$, CP é o comprimento do prefixo cíclico e N_p é o número de percursos considerado no canal, $\mathbf{x} \in \mathbb{C}^{(K+CP)\times 1}$, $\mathbf{h} \in \mathbb{R}^{(N_p)\times(1)}$ é a resposta ao impulso do

canal, $*$ é a operação de convolução e $\omega \in \mathbb{C}^{(K+CP+N_p-1)\times 1}$ é o ruído aditivo Gaussiano branco (AWGN). O sinal após o quantizador assumirá a forma $\mathbf{y}_q = \mathcal{Q}_c(\mathbf{y})$, onde $\mathcal{Q}_c(.)$ denota a função elementar de mapeamento do quantizador de valor complexo, que compreende dois quantizadores paralelos de valor real $\mathcal{Q}(.)$ que quantizam independentemente as partes real e imaginária de cada amostra de entrada analógica [8].

III. Prototipagem do sistema Q-OFDM com SDRs

Nesta Seção é detalhada a implementação do sistema usando as placas de interface HackRF One, RTL-SDR e a ferramenta de código aberto GNU Radio por meio de sua interface gráfica GNU Radio Companion (GRC). Para isso, esta Seção está dividida em três subseções: características do hardware utilizado [III-A], desenvolvimento do bloco do conversor A/D de baixa resolução [III-B] e, finalmente, descrição dos projetos do transmissor e do receptor [III-C].

A. Hardware utilizado

Os SDRs selecionados para este trabalho foram o HackRF One [9] e o RTL-SDR Blog V3 [10]. A partir das informações sobre estes dispositivos mostradas na Tabela I, nota-se que há uma janela de compatibilidade entre os SDRs em relação aos valores da frequência de operação e da taxa de amostragem.

TABELA I: Características básicas dos SDRs utilizados.

	HackRF One	RTL-SDR Blog V3
Comunicação	half duplex	recepção simplex
Frequência de operação	1 MHz - 6 GHz	500 kHz - 1766 MHz
Resolução do conversor	A/D - 8 bits	A/D - 8 bits
	D/A - 8 bits	
Taxa de amostragem	20 MS/s	2,4 MS/s

A taxa de amostragem escolhida foi de $1,8\,MS/s$ (*Mega-Samples/seg*), já a frequência de operação foi dependente da antena utilizada.

As antenas usadas com os SDRs transmissor e receptor são do tipo monopolo telescópica com conector SMA. No caso do HackRF One, que atuou como transmissor, além da antena também foi utilizado um DC-Blocker, a fim de fornecer para a antena um sinal puramente AC.

A performance da antena escolhida foi caracterizada a partir dos parâmetros medidos utilizando um VNA (Vector Network Analyzer), que é um instrumento para análise de redes que mede a resposta em amplitude e em fase da mesma.

O VNA usado foi o NanoVNA-H Rev3.6 [11] e com ele diversas medidas foram feitas nas antenas. Por meio do software NanoVNASaver [12], os dados puderam ser analisados em um computador e os resultados exibidos em sua tela, dos quais dois parâmetros medidos (VSWR e Perda de Retorno) são mostrados na Fig. 1. A partir desta análise a frequência de operação escolhida foi $410\,MHz$.

Os computadores escolhidos para este trabalho foram o SBC (Single Board Computer) Raspberry Pi 4 [13], que junto com o HackRF One formou o sistema SDR transmissor; e o notebook Lenovo Y510P [14], que somado ao RTL-SDR Blog V3 formou o sistema SDR receptor. Em ambos foi instalado a versão 3.7 do GNU Radio.

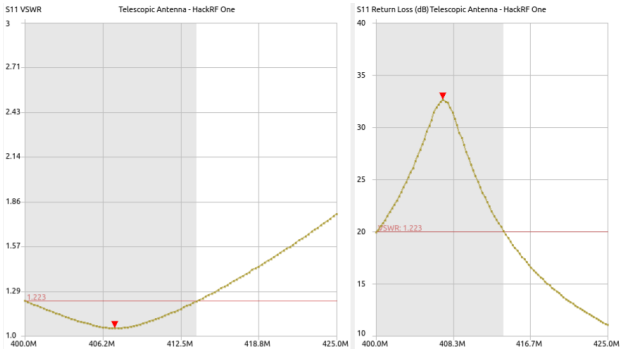

Fig. 1: Análise da antena entre 400 e $425\,MHz$ através do software NanoVNASaver.

TABELA II: Especificações dos computadores utilizados.

Computador		Raspberry Pi 4
CPU	processador	Broadcom BCM2711 (ARM Cortex-A72)
	núcleos	4
	frequência	1,5 GHz
Memória RAM		4GB LPDDR4 - 3200 MHz SDRAM
Sistema operacional		Raspberry Pi OS Lite (64-bit, 24/08/2020)

Computador		Lenovo Y510P
CPU	processador	Intel® Core™ i7-4700MQ
	núcleos	8
	frequênciaa	3,4 GHz
Memória RAM		8 GB DDR3 - 1600 MHz SODIMM
Sistema operacional		Lubuntu 18.04.5 LTS (64-bit)

A partir das informações da Tabela II nota-se que o notebook tem mais poder de processamento do que o Raspberry Pi, e por isso ele foi escolhido para a recepção, onde houve uma demanda maior do hardware. A Fig. 2 mostra a instalação completa do sistema de comunicação SDR utilizado neste trabalho. Do lado esquerdo observa-se o sistema transmissor e do lado direito, separado por $1\,m$ de distância, o sistema receptor.

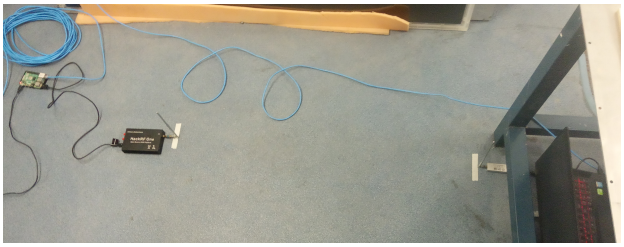

Fig. 2: Protótipo do Link SDR analisado.

B. Bloco do conversor A/D de baixa resolução

Nessa Seção, será descrito o Conversor Analógico Digital (ADC - *Analog-to-Digital Converter*) implementado neste trabalho e suas principais características. A escolha do quantizador linear *Mid-Riser*, se deve a sua ampla utilização, fácil implementação e bom desempenho [15]. Este quantizador divide o sinal a ser mapeado em níveis de quantização equidistantes, com um passo de tamanho Δ e número total de níveis de quantização (V) dado por $V = 2^b$, onde b é o número de bits de quantização utilizado no ADC. Para quantizadores *Mid-Riser* uniformes e simétricos, os níveis de saída do quantizador

são dados pela equação

$$v_i = \frac{-V\Delta}{2} + (i - \frac{1}{2})\Delta. \quad (2)$$

Os limites de entrada do quantizador são definidos por $\tau_1 = -\infty$, $\tau_{L+1} = \infty$ e $\tau_i = v_i + \frac{\Delta}{2}$, para $i = 2, 3, ..., V$ [15]. Portanto, para um sinal de entrada y discreto, a caracterização desse quantizador é dada pela equação (3) e ilustrado na Fig. 3.

$$Q(y[n]) \begin{cases} v_1, & y \leq \tau_1, \\ v_i, & \tau_{i-1} < y \leq \tau_i, \\ v_L, & y > \tau_{L+1}. \end{cases} \quad (3)$$

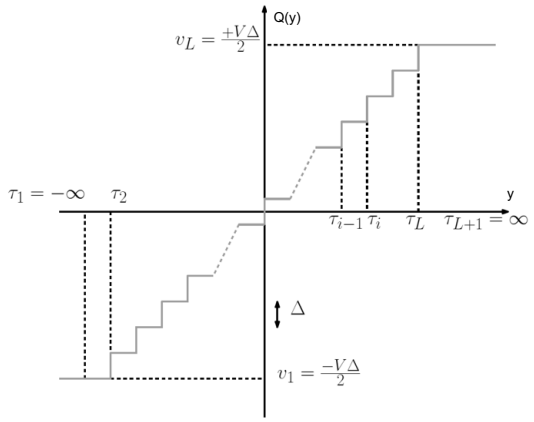

Fig. 3: Caracterização do quantizador *Mid-Riser* uniforme e simétrico.

A Fig. 4 mostra o projeto no GRC do bloco do conversor A/D de baixa resolução implementado pelos autores. Nota-

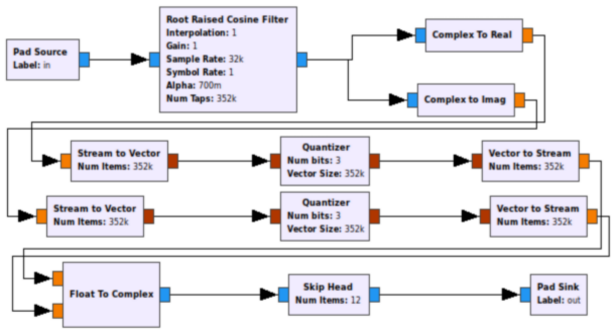

Fig. 4: Diagrama em blocos do conversor A/D de baixa resolução.

se que este bloco é composto, na verdade, por outros blocos, o que a documentação do GNU Radio classifica como um "Hierarchical Block" [16]. O fluxo do sinal passa por dois blocos principais: o "Root Raised Cosine Filter", que tem a função de simular uma conversão D/A; e o "Quantizer", que faz a quantização do sinal. Este último foi criado pelos autores a partir de um "Embedded Python Block" [17], um tipo especial de bloco do GNU Radio que permite que o usuário faça a sua customização por meio de um script em Python.

Desse modo, o bloco do conversor A/D de baixa resolução ficou disponível localmente na biblioteca do GRC com o nome de "Low Resolution Converter" e foi utilizado no projeto do receptor com diferentes bits de resolução.

C. Projetos do transmissor e do receptor

Os códigos e os projetos do GNU Radio utilizados neste trabalho estão disponíveis publicamente na plataforma GitHub [18]. O projeto do transmissor é mostrado na Fig. 5. Há o emprego de blocos específicos para cada etapa de uma transmissão OFDM e, propositalmente, não é utilizado o recurso de um código corretor de erros. A interface com o HackRF One é feita por meio do bloco "osmocom Sink". A

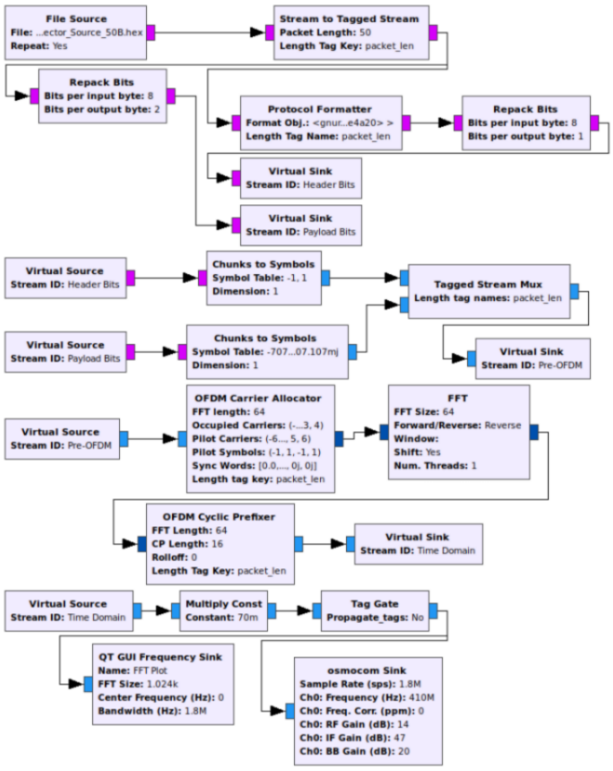

Fig. 5: Diagrama em blocos do transmissor.

transmissão consiste no envio contínuo de um arquivo de $50\,B$ de tamanho composto por bits gerados aleatoriamente.

A Fig. 6 mostra o projeto do receptor e a configuração que permite a coleta de dados de três eventos distintos: a potência média do sinal, a estimativa da performance do conversor A/D de baixa resolução e a performance do SDR receptor. A informação referente a potência média do sinal é de responsabilidade do bloco "Probe Avg Magˆ 2". Por meio do bloco "Low Resolution Converter" há a emulação do comportamento de um conversor A/D e ao desabilitá-lo do projeto, os dados coletados se referem diretamente à atuação do conversor A/D de 8 bits do RTL-SD Blog V3. De modo semelhante ao projeto do transmissor, são usados blocos específicos para cada etapa de uma recepção OFDM e a interface com o SDR é feita pelo bloco "osmocom Source". Além disso, houve também a adição de ruído gaussiano por meio do bloco "Noise Source".

As diversas medidas realizadas consistiram na coleta de dados na recepção para diferentes níveis de ruído. Os arquivos

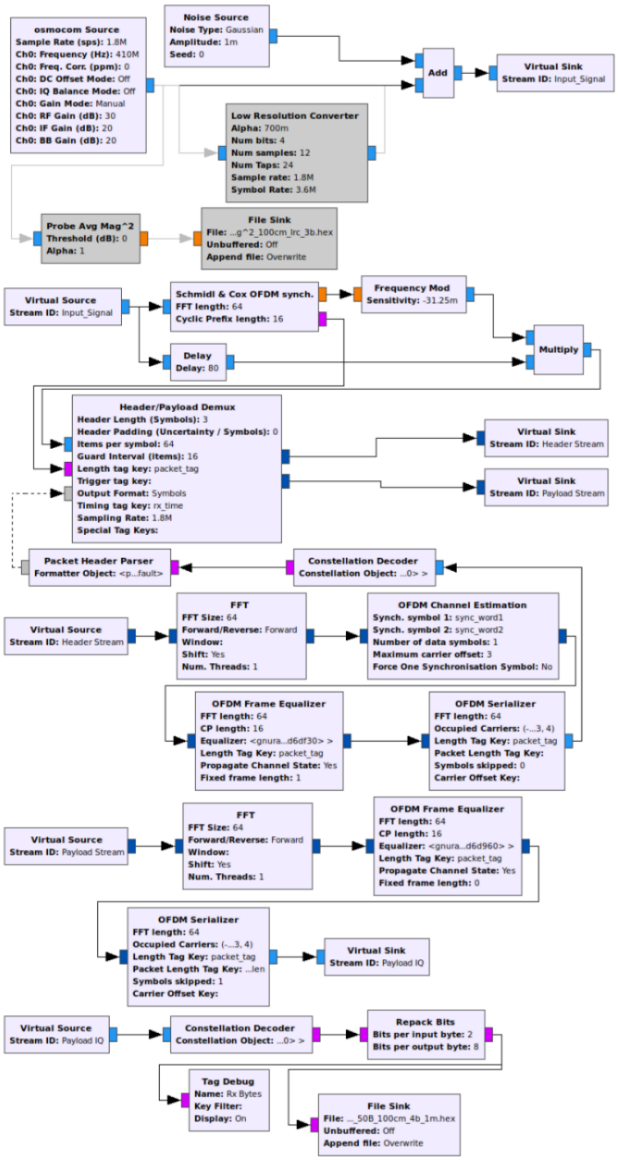

Fig. 6: Diagrama em blocos do receptor.

resultantes dessas medidas foram comparados com o arquivo transmitido. Desse modo, sabendo os níveis de ruído e a potência média do sinal foi possível o cálculo da SNR (Signal to Noise Ratio) e o levantamento das curvas de BER (Bit Error Rate) mostradas na Seção IV. As medidas foram realizadas, ainda, com as distâncias de 1 e 2 metros entre as antenas.

IV. Resultados

Os resultados da performance do sistema são mostrados nos gráficos das Figs. 7 e 8. Nestes gráficos a performance do SDR receptor é indicada pela curva "Hardware", ou seja, trata-se do comportamento do conversor A/D do RTL-SDR Blog V3 demonstrado pela execução do projeto B1_Rx. Todas as outras curvas são resultados para diferentes bits de resolução no bloco "Low Resolution Converter" do projeto B2_Rx, o que é indicado na legenda dos gráficos pelo termo "LRC" seguido do número de bits utilizado em cada medição.

Em ambos os gráficos o comportamento das curvas com um bit de quantização ("LRC 1b") e dois bits de quantização ("LRC 2b") apresentam uma BER muito alta e praticamente constante próxima de $0,5$. Isso era esperado devido a baixíssima resolução do conversor A/D, situação onde há um forte ruído de quantização. As curvas com 7 bits de quantização ("LRC 7b") e oito bits de quantização ("LRC 8b") também apresentam alta taxa de erro de bit. Os altos valores de BER das curvas "LRC 7b" e "LRC 8b", se devem ao *overrun* do hardware, isto é, um problema que ocorre quando o *stream* de dados de entrada produz mais dados do que o computador é capaz de processar.

Também em ambos os gráficos, a partir de 3 bits de resolução a performance do sistema começa a melhorar, com as curvas "LRC 3b", "LRC 4b", 'LRC 5b" e "LRC 6b" se aproximando da curva "Hardware". Entretanto, embora as curvas "LRC 5b" e "LRC 6b" se aproximem da curva "Hardware", elas têm um comportamento errático, que indica um princípio do efeito do *overrun*.

Outra tendência esperada, pelo menos teoricamente, é que com o aumento da relação sinal-ruído a BER mantenha o seu decaimento. Porém o que se observa nos gráficos é que após um determinado valor de SNR, uma melhora da relação sinal-ruído não implica em uma queda da taxa de erro de bit e as curvas se aproximam de um platô. Isso se deve ao nível do piso do ruído do canal e do próprio ruído térmico dos hardwares SDR.

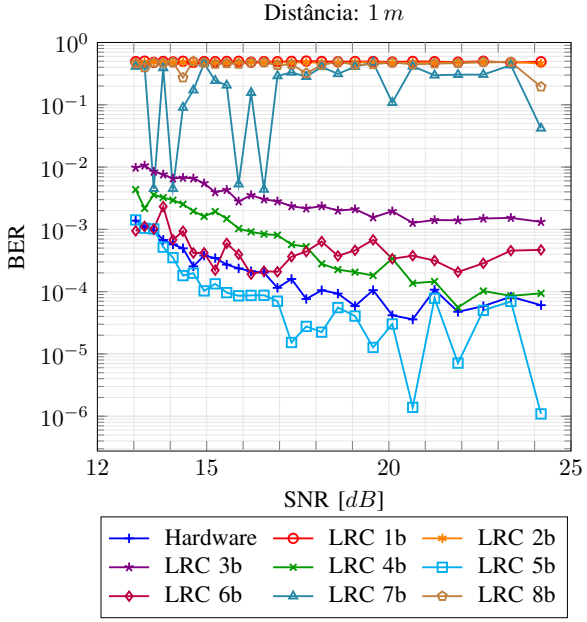

Fig. 7: Performance do sistema com $1\,m$ de distância entre antenas.

No gráfico de desempenho do sistema com 1 metro de distância entre as antenas (Fig. 7) as curvas se iniciam em $13,05\,dB$ de SNR (pior relação sinal-ruído possível de medida) e terminam em $24,18\,dB$ (relação sinal-ruído onde se observa o início do platô). A curva "LRC 4b" se inicia com uma BER próxima de 4×10^{-3}. A partir daí ela tem um decaimento que se aproxima da curva "Hardware" e a

partir de $20\,dB$ de SNR ambas convergem para uma BER de aproximadamente 9×10^{-5}.

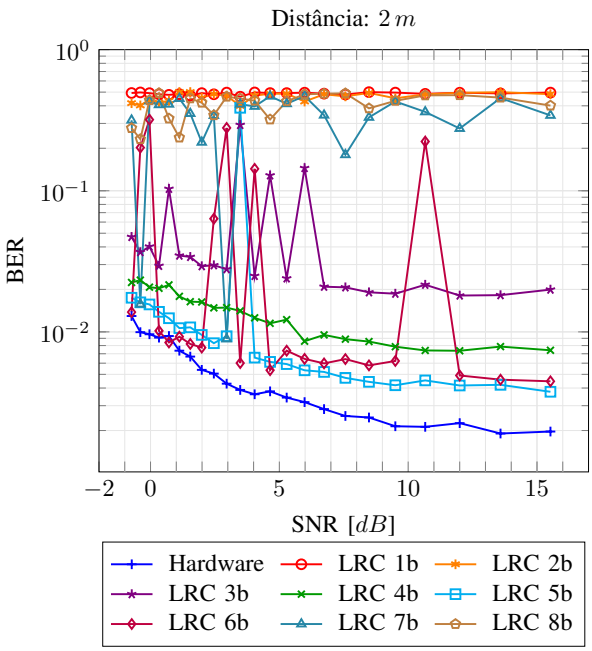

Fig. 8: Performance do sistema com $2\,m$ de distância entre antenas.

No gráfico da performance do sistema com 2 metros de distância entre as antenas (Fig. 8) a janela de relação sinal-ruído observada é diferente, justamente pelo maior espaçamento entre as antenas. As curvas se iniciam com uma SNR negativa em $-0,73\,dB$, o que indica que o nível de ruído é maior do que a potência do sinal, e terminam em $15,52\,dB$. Notam-se vários picos positivos de BER em diversas curvas, especialmente em regiões de baixa SNR. Esse fenômeno pode ter sido causado por uma maior interferência destrutiva oriunda das reflexões do sinal em múltiplos percursos do canal, devido a maior distância entre as antenas; ou alguma fonte de ruído impulsivo de algum equipamento na rede elétrica durante os testes. Uma vez que o experimento foi realizado em uma laboratório de Física sem os devidos controles de acesso e acionamento de equipamentos.

A curva "LRC 4b" tem uma BER inicial próxima de 2×10^{-2} e se aproxima assintoticamente de 7×10^{-3}. Portanto, nos dois cenários a curva "LRC 4b" apresentou um comportamento mais estável com uma performance razoável, isto é, com uma taxa de erro de bit próxima ou similar à taxa performada pelo conversor A/D do receptor. Considerando que a potência gasta no conversor A/D é proporcional ao número de bits de resolução, segundo a expressão $P_{ADC} \propto 2^{2b}$ [19], onde b é o número de bits, então uma redução de 8 para 4 bits representaria uma economia de energia estimada em $99,6\%$.

V. Conclusão

A partir do trabalho de prototipagem apresentado nesta pesquisa, é seguro afirmar que há fundamento para criação de um hardware crítico para sistemas que requerem uma grande eficiência energética, isto é, um conversor A/D com menos de 8 bits de resolução. É possível reforçar a importância do assunto abordado, visto que o mesmo pode impactar fortemente a prototipagem de sistemas rádio em laboratório. O uso de rádios definidos por software (SDRs) permite uma flexibilidade para testar diferentes sistemas de comunicação via rádio com o mesmo dispositivo. E embora SDRs, como USRP, já sejam amplamente usados em pesquisa, este trabalho mostrou que existem no mercado dispositivos de baixo custo que podem cumprir essa função. Os conteúdos aqui apresentados demonstram que muitas outras pesquisas ainda podem ser realizadas sobre prototipagem com uso de SDRs de baixo custo, devido à importância do tema e inúmeras contribuições para o meio acadêmico. Como exemplo de um trabalho futuro, pode-se considerar a implementação de um receptor no GNU Radio que leve em conta o ruído de quantização e assim melhorar a performance do sistema apresentado mesmo funcionando em baixa resolução.

Referências

[1] Y. Mehmood, N. Haider, M. Imran, A. Timm-Giel, M. Guizani, *M2M Communications in 5G: State-of-the-Art Architecture, Recent Advances, and Research Challenges*, IEEE Communications Magazine, 2017.
[2] M. Braun, *OFDM – You're looking fantastic these days*, GNU Radio Conference, Boston, 2013.
[3] X. Xiong, W. Tong, L. Hang, Z. Kan, *Implementation and performance evaluation of LECIM for 5G M2M applications with SDR*. Proceedings of 2014 IEEE Globecom Workshops.
[4] https://www.ettus.com/all-products/un200-kit. Visitado em 03/06/2022.
[5] G. Silva, J. T. Dias, *Equalizador LRA-MMSE para sistemas OQ-OFDM de baixa resolução*, SBrT2020.
[6] E. O. de Souza, J. T. Dias, D. Gonçalves, *Implementação e Análise de Desempenho de um Sistema OFDM com Hardware de Baixo Custo*, SBrT2021.
[7] N. Marchetti, M. I. Rahman, S. Kumar, R. Prasad, "New Directions in Wireless Communications Research", cap. 2, *OFDM-Principles and Challenges*, 2009.
[8] C. studer and G. Durisi, *Quantized Massive MU-MIMO-OFDM Uplink*. IEEE Transactions on Communications, Vol. 64, no. 6, 2016.
[9] https://greatscottgadgets.com/hackrf/one/. Visitado em 03/06/2022.
[10] https://www.rtl-sdr.com/about-rtl-sdr/. Visitado em 03/06/2022.
[11] https://nanovna.com/. Visitado em 03/06/2022.
[12] https://nanovna.com/?page_id=90. Visitado em 03/06/2022.
[13] https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. Visitado em 03/06/2022.
[14] https://www.lenovo.com/in/en/laptops/lenovo/y-series/y510p/. Visitado em 03/06/2022.
[15] A. Gersho, R. M. Gray, *Vector Quantization and Signal Compression*. Springer, 1a. edição, 1992.
[16] https://wiki.gnuradio.org/index.php/BlocksCodingGuide#Hierarchical_Block. Visitado em 03/06/2022.
[17] https://wiki.gnuradio.org/index.php/Embedded_Python_Block. Visitado em 03/06/2022.
[18] https://github.com/ederOlive/SBrT2022.git
[19] Mezghani, A., Nossek, J. A., *How to choose the ADC resolution for short range low power communication?*., Proceedings of 2010 IEEE International Symposium on Circuits and Systems.