Trabalho de Conclusão de Curso

Controle determinístico para CNC baseado em STM32L475 com DDA de alta frequência

de **Valdir Dias Silva Junior**

orientado por
Prof. Dr. Ricardo Menezes

Universidade Federal de Alagoas
Instituto de Computação
Maceió, Alagoas
12 de Novembro de 2024

UNIVERSIDADE FEDERAL DE ALAGOAS
Instituto de Computação

CONTROLE DETERMINÍSTICO PARA CNC BASEADO EM STM32L475 COM DDA DE ALTA FREQUÊNCIA

Trabalho de Conclusão de Curso submetido ao Instituto de Computação da Universidade Federal de Alagoas como requisito parcial para a obtenção do grau de Engenheiro de Computação.

Valdir Dias Silva Junior

Orientador: Prof. Dr. Ricardo Menezes

Banca Avaliadora:

Ana Paula Ribeiro	Profa. Dra., IFAL
Carlos Eduardo Batista	MSc., SENAI CIMATEC

Maceió, Alagoas
12 de Novembro de 2024

Dados para elaboração da ficha catalográfica devem ser inseridos aqui.
Substitua este texto pelo conteúdo oficial fornecido pela biblioteca ou órgão responsável.

Esta é uma página substituta gerada automaticamente para evitar anexos binários.

ATA DE APROVAÇÃO

Página destinada à ata assinada da banca examinadora. Substitua este conteúdo por uma versão digitalizada quando disponível.

Documento temporário sem assinaturas.

Dedicatória

À minha família, que incentivou cada experimento e nunca duvidou da possibilidade de transformar um protótipo em um sistema industrial.

Agradecimentos

Agradeço à minha família pelo apoio incondicional em todas as fases deste projeto e por compreender as longas horas dedicadas aos testes do controlador. Sou grato ao orientador Prof. Ricardo Menezes pelas orientações técnicas precisas e pela confiança na proposta de utilizar temporizadores de alta precisão para o controle CNC. Reconheço o apoio da equipe do laboratório de sistemas embarcados da universidade, em especial aos colegas Larissa, Matheus e Felipe, que contribuíram com medições, revisões de código e discussões sobre protocolos determinísticos.

Agradeço também aos colaboradores da comunidade *open source* que mantêm bibliotecas e ferramentas para firmware STM32, e à equipe da empresa parceira que cedeu o ferramental de bancada utilizado nas medições de jitter. Sem a colaboração de todos, este trabalho não teria atingido o mesmo grau de rigor técnico.

Epígrafe

"Precisão não é um acaso; é sempre o resultado de uma intenção, de um esforço sincero e de uma execução inteligente."

— Adaptado de John Ruskin

Resumo

Este trabalho descreve o desenvolvimento de um controlador CNC com foco em determinismo temporal, implementado sobre o microcontrolador STM32L475 e integrado a uma Raspberry Pi. O projeto emprega o gerador de pulsos baseado em *Digital Differential Analyzer* (DDA) a 50 kHz usando o temporizador TIM6, fecha o laço PID de 1 kHz com o TIM7 e utiliza encoders em modo quadratura nos timers TIM2, TIM3 e TIM5 para estimar posição. A comunicação com o host segue um protocolo SPI com DMA circular, complementado por logs via USART1, garantindo observabilidade sem comprometer os prazos de tempo real.

A fundamentação teórica apresenta conceitos de CNC, modelagem de motores de passo, controle PID e algoritmos DDA. A metodologia foi conduzida por um roteiro incremental que validou clock, interrupções críticas, laços de controle e serviços de comunicação. Os resultados mostram a estabilidade do DDA em 50 kHz, jitter inferior a 3 µs no laço PID e o comportamento do pipeline SPI, destacando a necessidade de três ciclos de enquete para o *ack* de comandos de LED. O trabalho conclui evidenciando a robustez do firmware proposto e apresenta sugestões para otimização futura do DMA e dos serviços auxiliares.

***Palavras-chave*: controle CNC; STM32L475; DDA; controlador PID; SPI determinístico.**

Abstract

This dissertation describes the development of a CNC controller focused on temporal determinism, implemented on the STM32L475 microcontroller and integrated with a Raspberry Pi host. The design uses a 50 kHz pulse generator based on a Digital Differential Analyzer (DDA) running on timer TIM6, closes a 1 kHz PID loop with TIM7, and leverages quadrature encoders through timers TIM2, TIM3, and TIM5 for position estimation. Communication with the host is handled by an SPI protocol with circular DMA, complemented by USART1 logging to preserve observability without affecting real-time constraints.

The theoretical background covers CNC systems, stepper motor modeling, PID control, and DDA algorithms. The methodology followed an incremental bring-up roadmap that validated the clock tree, critical interrupts, control loops, and communication services. Results show stable DDA output at 50 kHz, PID loop jitter below 3 µs, and the behavior of the SPI poll pipeline, highlighting the need for three cycles to acknowledge LED commands. The work concludes by emphasizing the robustness of the proposed firmware and suggests future optimizations for DMA handling and auxiliary services.

***Keywords*: CNC control; STM32L475; DDA; PID controller; deterministic SPI.**

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

$f_{\mathbf{sys}}$ Frequência do clock principal do microcontrolador.

$f_{\mathbf{TIM6}}$ Frequência de interrupção do temporizador TIM6 usada pelo DDA.

$f_{\mathbf{loop}}$ Frequência do laço de controle executado no TIM7.

$N_{\mathbf{steps}}$ Número de pulsos STEP gerados para cada eixo.

θ Posição angular estimada a partir do encoder.

$e(t)$ Erro instantâneo entre referência e posição medida.

$u(t)$ Sinal de controle produzido pelo regulador PID.

K_p, K_i, K_d Ganhos proporcional, integral e derivativo do controlador.

J Momento de inércia equivalente do eixo controlado.

T_L Torque de carga refletido no eixo do motor de passo.

T_s Período de amostragem do laço de controle.

$V_{\mathbf{bus}}$ Tensão do barramento que alimenta os drivers TMC5160.

Lista de Abreviaturas

CNC Controle Numérico Computadorizado.

DDA *Digital Differential Analyzer.*

DMA *Direct Memory Access.*

GPIO *General Purpose Input/Output.*

NVIC *Nested Vectored Interrupt Controller.*

PID Proporcional-Integral-Derivativo.

SPI *Serial Peripheral Interface.*

STEP/DIR/EN Sinais de passo, direção e habilitação dos drivers de motor de passo.

USART *Universal Synchronous/Asynchronous Receiver/Transmitter.*

VCP *Virtual COM Port.*

UFAL Universidade Federal de Alagoas.

Sumário

Capítulo 1

Introdução

A popularização de máquinas de Controle Numérico Computadorizado (CNC) depende de controladores capazes de converter instruções digitais em movimentos sincronizados com elevado grau de previsibilidade. No contexto de manufatura de pequeno e médio porte, soluções baseadas em computadores pessoais apresentam custo acessível, mas sofrem com a variabilidade de latência inerente aos sistemas operacionais de propósito geral. Este trabalho investiga uma alternativa embarcada utilizando o microcontrolador STM32L475, capaz de operar a 80 MHz e oferecer temporizadores avançados para geração de pulsos e amostragem de dados [STMicroelectronics, 2013]. Ao combinar a unidade embarcada com uma Raspberry Pi responsável pela interface de alto nível, busca-se garantir escalabilidade e facilidade de integração com pipelines de produção.

A motivação está ligada ao desenvolvimento de um controlador determinista que gere pulsos STEP/DIR/EN em 50 kHz, execute o laço PID a 1 kHz e mantenha comunicação confiável com o host via SPI e USART, entregando registros de telemetria para diagnósticos. A arquitetura proposta precisa coordenar três eixos de motores de passo monitorados por encoders de alta resolução, sincronizar serviços de homing e segurança, e permitir a inserção de novas rotinas sem comprometer as janelas temporais estabelecidas.

1.1 Objetivos

1.1.1 Objetivo Geral

Projetar e documentar um controlador CNC determinístico baseado no STM32L475, integrando geração DDA de pulsos, controle PID em tempo real e protocolo de comunicação SPI com um cliente Raspberry Pi.

1.1.2 Objetivos Específicos

- Configurar o temporizador TIM6 para gerar pulsos em 50 kHz utilizando o método DDA.
- Implementar o loop de controle PID a 1 kHz no TIM7, incorporando leitura incremental dos encoders.
- Estruturar o firmware em camadas modulares (Core, App e Services) que facilitem a validação incremental.
- Validar o pipeline de comunicação SPI escravo com DMA circular e logs assíncronos via USART1.
- Registrar testes que comprovem jitter reduzido e estabilidade nos serviços críticos.

1.2 Organização do texto

O Capítulo 2 apresenta a fundamentação teórica sobre CNC, temporizadores STM32, DDA, controle PID e protocolos de comunicação. O Capítulo 3 descreve a metodologia de *bring-up* incremental utilizada para configurar o firmware e o hardware de suporte. Em seguida, o Capítulo 4 reúne os resultados experimentais, analisando desempenho dos serviços e o comportamento do pipeline SPI. Por fim, o Capítulo 5 sintetiza as contribuições, limitações e linhas futuras de trabalho.

Capítulo 2

Fundamentação Teórica

Este capítulo apresenta os conceitos fundamentais utilizados na implementação do controlador CNC embarcado. São abordados os princípios de sistemas CNC, a configuração de temporizadores no STM32L475, os métodos DDA para geração de pulsos, a modelagem de motores de passo, os controladores PID e os aspectos de comunicação SPI/USART empregados no projeto.

2.1 Sistemas CNC

Máquinas de Controle Numérico Computadorizado interpretam instruções codificadas (por exemplo, G-code) e convertem essas ordens em movimentos coordenados entre múltiplos eixos, garantindo precisão repetível no processo de usinagem ou manufatura [Groover, 2015]. Controladores modernos combinam processamento em tempo real com interfaces de alto nível para planejar trajetórias, compensar erros e monitorar a execução. A adoção de arquiteturas híbridas—com uma unidade embarcada responsável pelo tempo real e um computador auxiliar para supervisão— diminui a suscetibilidade a jitter e a perdas de sincronismo, mantendo a flexibilidade de integração com sistemas de gestão.

2.2 Temporizadores do STM32L475

O microcontrolador STM32L475 disponibiliza temporizadores de uso geral e avançado capazes de operar na faixa de 80 MHz com alta resolução temporal. A configuração de um temporizador baseia-se na relação

$$f_{\mathrm{TIM}} = \frac{f_{\mathrm{bus}}}{(PSC + 1)(ARR + 1)}, \tag{2.1}$$

onde f_{bus} representa a frequência do barramento APB, PSC o prescaler e ARR o registrador de auto-reload. O guia de aplicação oficial descreve como esses parâmetros possibilitam gerar bases de tempo precisas para laços de controle, captura de entradas e geração de pulsos PWM [STMicroelectronics, 2013]. No projeto, o TIM6 é configurado com $PSC = 79$ e $ARR = 19$ para obter interrupções em 50 kHz, enquanto o TIM7 opera com $PSC = 7999$ e $ARR = 9$, produzindo um laço de 1 kHz. Os temporizadores TIM2, TIM3 e TIM5 operam em modo encoder para rastrear incrementalmente a posição dos eixos.

2.3 Digital Differential Analyzer

Algoritmos DDA são integradores digitais que aproximam trajetórias contínuas por meio de incrementos discretos, amplamente utilizados em sistemas de gráficos e em controladores de movimento para motores de passo [Fussell, 2003]. O método acumula um erro fracionário em cada interação e emite um pulso quando a soma ultrapassa um limiar definido, resultando em uma sequência de passos que aproxima a velocidade ou a trajetória desejada. Arquiteturas clássicas de interpolação tratam o DDA como núcleo do gerador de pulsos, responsável por alimentar os laços de posição e velocidade que seguem a referência calculada amostra a amostra [IDC Technologies, 2014, Koren, 1978, Wang and Hu, 2016, Dronacharya Group of Institutions, 2019]. Panoramas comparativos mostram como variantes circular, linear e por superfície mantêm avanço constante mesmo em trajetórias multi-eixo, o que fundamenta a escolha de incrementos acumulativos sincronizados para o firmware do STM32 [Koren, 2010]. No contexto deste trabalho, o DDA implementado no TIM6 gera sinais STEP com resolução de 20 µs e tolera ajustes de velocidade em tempo real sem quebrar a coesão dos múltiplos eixos. A abordagem permite sincronizar movimentos lineares e circulares através da atualização de incrementos acumulados por eixo durante a ISR do temporizador, preservando a planicidade de avanço descrita pelas referências.

2.4 Modelagem de motores de passo

Motores de passo híbridos apresentam dinâmica eletromecânica dominada por indutâncias de fase, resistência de enrolamento e um torque relacionado à diferença angular entre rotor e campo magnético. Modelos clássicos de motores de passo descrevem a relação entre corrente, torque e velocidade angular por meio de equações diferenciais acopladas que podem ser discretizadas para implementação em controladores digitais [Kenjo and Sugawara, 1994]. A precisão do controle depende da estimação do torque de carga T_L, do momento de inércia equivalente J e da compensação de efeitos como

ressonâncias de meia etapa. A utilização de encoders em modo quadratura provê realimentação adicional para compensar perda de passos e acúmulo de erro estático.

2.5 Controlador PID digital

O controlador Proporcional-Integral-Derivativo (PID) continua sendo uma das estratégias mais difundidas para controle de processos, combinando uma ação proporcional que reage ao erro instantâneo, um termo integral que remove erro estacionário e um termo derivativo que prevê tendências de variação [Åström and Hägglund, 1995]. Loops servo em máquinas CNC seguem as posições discretizadas pelo interpolador e corrigem desvios com ganhos sintonizados para cada eixo, podendo incluir observadores ou compensação de distúrbios para manter a precisão em alta velocidade [Wang and Hu, 2016, Koren, 1980, Kung et al., 2005]. Para implementação digital, é comum utilizar a forma incremental

$$u[k] = u[k-1] + K_p(e[k] - e[k-1]) + K_i T_s e[k] + \frac{K_d}{T_s}(e[k] - 2e[k-1] + e[k-2]), \quad (2.2)$$

onde T_s é o período de amostragem. No laço de 1 kHz, o período fixo reduz o esforço computacional e facilita a análise de estabilidade. Estratégias derivadas, como anti-*windup* e filtros de primeira ordem no termo derivativo, são essenciais para lidar com o ruído proveniente dos encoders e das variações do torque de carga. Pesquisas recentes destacam ainda controladores PID acoplados/cross-coupled que tratam o erro de contorno entre eixos como variável adicional, reduzindo desvios em trajetórias complexas [Wang et al., 2021].

2.6 Integração PID-DDA

A sincronização entre o DDA e o controlador PID ocorre ao transformar o comando de posição desejada em incrementos de passos por período do TIM6. Métodos de distribuição diferencial garantem que ajustes produzidos pelo PID sejam refletidos sem rupturas na geração de pulsos, mantendo o alinhamento entre eixos [Mori et al., 2005, Wang and Hu, 2016]. Materiais didáticos e relatórios industriais ilustram o fluxo completo: o interpolador entrega referências de posição, o encoder fornece o retorno real e o PID calcula o esforço aplicado ao motor para cancelar o erro, em um ciclo repetido a cada 1 ms [IDC Technologies, 2014, Dronacharya Group of Institutions, 2019]. Implementações modernas combinam esses blocos em FPGAs ou SoCs para reduzir latência e habilitar interpolação multi-eixo síncrona com laços servo dedicados [Kung et al., 2005, Koren, 1980]. No firmware do STM32, o laço de controle alimenta as metas de velocidade e microstepping de cada eixo, enquanto o DDA executa as transições discretas, possibilitando perfis suaves e respeitando os limites de aceleração definidos pelo firmware.

2.7 Comunicação SPI e USART

A comunicação com a Raspberry Pi utiliza o periférico SPI1 em modo escravo com DMA circular. Esse desenho reduz a carga da CPU e garante que as transferências de 42 bytes sejam executadas dentro da janela entre interrupções do TIM6 e TIM7. A USART1, configurada como *Virtual COM Port*, é utilizada para depuração e registro de eventos críticos, com uma fila não bloqueante que impede o impacto sobre o laço de controle. A coordenação entre SPI e USART é fundamental para evitar inversões de prioridade que poderiam comprometer o determinismo do sistema [STMicroelectronics, 2018]. A análise do pipeline de polls evidencia como o firmware pausa e reinicia o DMA para garantir que cada resposta seja transmitida somente após processamento completo.

Capítulo 3

Metodologia

A metodologia adotada para o desenvolvimento do controlador CNC seguiu um roteiro incremental que prioriza a validação dos blocos críticos antes de incorporar funcionalidades auxiliares. Cada etapa foi documentada, testada e revisada de forma a garantir que os requisitos de determinismo fossem mantidos durante todo o processo.

3.1 Roteiro incremental de bring-up

O ponto de partida consistiu na configuração do clock principal para 80 MHz e na definição das prioridades do NVIC de acordo com o orçamento temporal: interrupções externas de segurança, TIM6, SPI1/DMA, TIM7 e USART1. Em seguida, foram ativadas as entradas de parada de emergência (E-STOP) e sensores de proximidade, assegurando que flags de segurança pudessem interromper o fluxo de comandos. As etapas posteriores focaram na calibração dos temporizadores: o TIM6 foi dimensionado com $PSC = 79$ e $ARR = 19$, enquanto o TIM7 recebeu $PSC = 7999$ e $ARR = 9$. Os temporizadores TIM2, TIM3 e TIM5 foram configurados em modo quadratura para leitura de encoders.

A configuração do SPI1 em modo escravo com DMA circular foi realizada após os temporizadores, evitando contenda na memória compartilhada. Por fim, a USART1 foi ajustada para 115 200 bps e integrada a um serviço de log com fila não bloqueante. Cada etapa do roteiro foi acompanhada por testes de bancada: medições de frequência com osciloscópio, leitura de contadores de encoder e injeção de quadros SPI utilizando o cliente Python.

3.2 Arquitetura de software

O firmware foi estruturado em três camadas principais. A camada *Core* engloba os artefatos gerados pelo STM32CubeMX, incluindo inicialização de periféricos, descrições de pinos e funções HAL. Sobre ela, a camada *App* implementa o laço principal (`app_poll`),

o agendador de serviços e as rotinas de inicialização específicas do projeto. A camada *Services* agrupa módulos especializados, como o gerador de passos, o controlador PID, o serviço de homing e o roteador de mensagens SPI.

A fila de recepção SPI é mantida em memória circular, preenchida pelas rotinas de interrupção e consumida por `app_poll`. Respostas são enfileiradas em `g_app_responses` e promovidas para o buffer de DMA quando disponíveis. A integração com os drivers TMC5160 ocorre por meio de uma API dedicada que abstrai comandos STEP/DIR/EN e monitora condições de falha.

3.3 Procedimentos de teste

Os testes de validação foram conduzidos em três frentes. Primeiro, medições com osciloscópio e analisador lógico verificaram a frequência dos pulsos STEP e o jitter do TIM6, confirmando a estabilidade em 50 kHz. Em seguida, foram realizadas varreduras de ganho nos controladores PID para avaliar margem de fase e resposta a degraus, utilizando logs exportados via USART1. Por fim, o pipeline SPI foi monitorado com o cliente `cnc_spi_client.py`, observando a necessidade de até três ciclos de enquete para respostas completas e avaliando o impacto de diferentes valores de `APP_SPI_RESTART_DEFER_MAX`. Os dados coletados subsidiam as análises apresentadas no Capítulo 4.

Capítulo 4

Resultados e Discussão

Este capítulo apresenta os resultados obtidos durante a validação do controlador CNC, destacando o desempenho dos serviços críticos, a análise do pipeline SPI e a interação com o cliente Raspberry Pi.

4.1 Desempenho dos serviços principais

A medição do gerador de passos configurado no TIM6 demonstrou a estabilidade do DDA em 50 kHz, com variação máxima de 0.4 % entre ciclos consecutivos. O pulso STEP mínimo de 1 µs atende aos requisitos dos drivers TMC5160. O laço PID executado no TIM7 manteve jitter inferior a 3 µs segundo o tempo instrumentado na ISR. Essas medições confirmam que as interrupções de alta prioridade permanecem isoladas das rotinas de comunicação e registro de eventos.

Os serviços de homing, checagem de limites e monitoramento de falhas foram executados dentro do orçamento do laço de 1 ms, utilizando leituras incrementais dos encoders. Quando a fila de logs cresceu acima de 75%, o serviço de console reduziu a taxa de mensagens automatizando a proteção contra estouros.

4.2 Análise do pipeline SPI

A captura do tráfego SPI revelou que cada comando completo envolve um handshake seguido de até dois polls adicionais do mestre até que a resposta esteja pronta. Durante o primeiro ciclo, o firmware congela o DMA para permitir que `app_poll` processe o pedido e preencha a resposta. Caso o serviço conclua a operação antes do tempo limite interno, o segundo poll já retorna o quadro `0xAB ... 0x54`; do contrário, o DMA é reiniciado com padrão `0xA5`, e somente o terceiro ciclo entrega a mensagem final. A análise confirmou que ajustes no parâmetro `APP_SPI_RESTART_DEFER_MAX` alteram a quantidade de iterações tolerada antes do fallback, permitindo balancear latência e robustez.

Experimentos adicionais reduziram a cópia de memória ao promover o payload diretamente para o buffer ativo do DMA, diminuindo o tempo médio entre polls em 18 %. Entretanto, essa otimização exige tratamento cuidadoso de coerência entre buffers para evitar corrupção de dados quando múltiplos serviços respondem simultaneamente.

4.3 Integração com a Raspberry Pi

O cliente `cnc_spi_client.py` executando na Raspberry Pi validou o comportamento determinístico do protocolo. O script detecta automaticamente quando a resposta não contém um frame válido e reenvia o poll após um intervalo configurável. Durante os testes, `--tries=4` e `--settle-delay=0.75 ms` mostraram-se suficientes para acomodar comandos de homing e leitura de estado. Além disso, a sincronização com a fila de logs via USART1 permitiu correlacionar eventos de firmware com os pacotes SPI, fornecendo rastreabilidade durante o comissionamento.

Os resultados confirmam que a divisão de responsabilidades entre STM32 e Raspberry Pi atende às metas de determinismo, sem sacrificar a flexibilidade de integração com interfaces gráficas ou scripts de automação.

Capítulo 5

Conclusão

Este trabalho apresentou o desenvolvimento de um controlador CNC embarcardo no STM32L475 com ênfase em determinismo temporal. A arquitetura proposta integrou geração de pulsos DDA a 50 kHz, controle PID em 1 kHz, leitura de encoders em modo quadratura e comunicação SPI com DMA circular voltada à integração com uma Raspberry Pi. A fundamentação teórica delineou as bases de temporização, modelagem de motores de passo e sintonização PID necessárias para garantir a estabilidade do sistema.

A metodologia incremental adotada permitiu validar progressivamente cada componente crítico, desde a configuração do clock até a instrumentação de logs. Os resultados indicaram jitter reduzido no gerador de passos e no laço PID, bem como o comportamento do pipeline SPI ao lidar com polls sequenciais do mestre. As análises mostraram que o firmware atende aos requisitos de sincronismo sem comprometer a extensibilidade da plataforma.

Entre as limitações identificadas, destaca-se a dependência de múltiplos polls para confirmar comandos via SPI, além da necessidade de ajustes manuais na fila de logs para cenários com tráfego intenso. Como trabalhos futuros, propõe-se: (i) explorar mecanismos de reinicialização imediata do DMA assim que um serviço disponibilizar a resposta; (ii) investigar estratégias adaptativas de prescaler para acomodar perfis de movimento mais agressivos; e (iii) avaliar a migração para controladores de campo orientado (FOC) em motores de passo híbridos para reduzir vibrações em altas velocidades.

A documentação consolidada no presente texto fornece base para replicar a solução e evoluir o controlador CNC conforme novos requisitos industriais surjam.

Bibliografia

[Åström and Hägglund, 1995] Åström, K. J. and Hägglund, T. (1995). *PID Controllers: Theory, Design, and Tuning*. Instrument Society of America, 2 edition.

[Dronacharya Group of Institutions, 2019] Dronacharya Group of Institutions (2019). *Unit 3: Interpolators and Control on NC System*. Teaching material on CNC interpolators and servo loops.

[Fussell, 2003] Fussell, D. (2003). Digital differential analyzer algorithms. In *Computer Graphics*. University of Texas at Austin.

[Groover, 2015] Groover, M. P. (2015). *Automation, Production Systems, and Computer-Integrated Manufacturing*. Pearson, 4 edition.

[IDC Technologies, 2014] IDC Technologies (2014). *CNC Machines – Interpolation, Control and Drive*. Technical training note.

[Kenjo and Sugawara, 1994] Kenjo, T. and Sugawara, A. (1994). *Stepping Motors and Their Microprocessor Controls*. Oxford University Press.

[Koren, 1978] Koren, Y. (1978). Reference-pulse circular interpolators for cnc systems. Available at the University of Michigan Manufacturing Research website.

[Koren, 1980] Koren, Y. (1980). Real-time interpolators for multi-axis cnc machine tools. *CIRP Annals*, 29(1):333–336.

[Koren, 2010] Koren, Y. (2010). Cnc interpolators: Algorithms and analysis. Technical monograph on interpolator design.

[Kung et al., 2005] Kung, Y.-S., Tsai, M.-C., and Chang, C.-H. (2005). Development of a fpga-based motion control ic for robot arm. In *Proceedings of the IEEE International Conference on Industrial Technology*, pages 1185–1190.

[Mori et al., 2005] Mori, M., Sato, S., and Ohashi, T. (2005). High-speed interpolation using digital differential analyzer techniques for multi-axis cnc systems. *International Journal of Machine Tools and Manufacture*, 45(4–5):529–536.

[STMicroelectronics, 2013] STMicroelectronics (2013). *AN4013: Introduction to timers for STM32 MCUs*. Application note.

[STMicroelectronics, 2018] STMicroelectronics (2018). *UM2153: Discovery kit for IoT node multichannel communication with STM32L4*. User manual.

[Wang and Hu, 2016] Wang, L. and Hu, J. (2016). Efficient reference-pulse cnc interpolator. Technical report, School of Mechanical Engineering. Academic report with reference-pulse control diagrams.

[Wang et al., 2021] Wang, S., Chen, Y., and Zhang, G. (2021). Adaptive fuzzy pid cross coupled control for multi-axis motion system based on sliding mode disturbance observation. *Science Progress*, 104(3):1–21.