# TI AFE7920 RF 트랜시버 제어를 위한 Latte 소프트웨어 종합 기술 보고서

## 섹션 1: 아키텍처 심층 분석: AFE7920 RF-샘플링 트랜시버

Texas Instruments(TI)의 AFE7920은 최신 무선 통신 인프라의 까다로운 요구사항을 충족시키기 위해 설계된 고성능 RF-샘플링 트랜시버입니다. 이 장치의 복잡성과 유연성을 완벽하게 활용하기 위해서는 제어 소프트웨어인 Latte의 역할을 이해하기에 앞서, 먼저 하드웨어 아키텍처에 대한 깊이 있는 이해가 선행되어야 합니다. AFE7920은 단순한 부품이 아닌, 칩 하나에 완전한 RF 서브시스템을 통합한 고도로 집적된 솔루션입니다.

### 1.1 핵심 시스템 온 칩(SoC) 역량: 4T4R+2FB의 강력한 성능

AFE7920의 핵심은 다중 채널을 지원하는 아키텍처에 있습니다. 이 장치는 4개의 RF 샘플링 송신(TX) 체인, 4개의 RF 샘플링 수신(RX) 체인, 그리고 2개의 RF 샘플링 디지털화 보조 체인(피드백 경로, FB)을 통합하고 있습니다.1 이러한 4T4R+2FB (4-Transmit, 4-Receive, 2-Feedback) 구성은 3G, 4G 및 5G 무선 기지국과 같은 고급 통신 시스템에 필수적입니다.1 특히, 2개의 독립적인 피드백 경로는 전력 증폭기(PA)의 비선형성을 보상하기 위한 디지털 전치 왜곡(Digital Pre-Distortion, DPD)과 같은 정교한 PA 선형화 기술을 구현하는 데 결정적인 역할을 합니다.4 이 장치는 매크로 원격 라디오 유닛(RRU), 대규모 다중 입출력(Massive MIMO)을 위한 능동 안테나 시스템(AAS), 스몰셀 기지국, 중계기 등 다양한 애플리케이션을 위해 설계되었습니다.3 AFE7920은 17mm x 17mm 크기의 FCBGA 패키지로 제공되어 고밀도 시스템 설계에 유리합니다.3

### 1.2 고속 데이터 변환 및 통합 디지털 신호 처리(DSP) 체인

AFE7920의 성능은 내장된 고속 데이터 컨버터와 통합 DSP 블록에서 비롯됩니다. 이 아키텍처는 기존의 아날로그 방식과 비교하여 높은 유연성과 성능을 제공합니다.

* **송신(TX) 경로**: 4개의 12-GSPS (Giga-Samples Per Second) 디지털-아날로그 변환기(DAC)를 탑재하고 있습니다. 각 송신 체인은 최대 1200 MHz의 결합 신호 대역폭을 지원하는 단일 또는 이중 디지털 상향 변환기(DUC)를 포함합니다. DAC 출력 단에는 40 dB 범위의 가변 이득 증폭기(TX DSA)가 있어 정밀한 출력 전력 제어가 가능합니다.1 12 GSPS라는 매우 높은 샘플링 속도는 많은 경우 전통적인 아날로그 상향 변환 단계를 생략하고 RF 신호를 직접 합성하는 것을 가능하게 합니다.
* **수신(RX) 경로**: 4개의 3-GSPS 아날로그-디지털 변환기(ADC)를 특징으로 합니다. 각 수신 체인에는 25 dB 범위의 디지털 스텝 감쇠기(DSA)와 최대 600 MHz의 결합 신호 대역폭을 제공하는 단일 또는 이중 디지털 하향 변환기(DDC)가 포함되어 있습니다.1
* **피드백(FB) 경로**: 2개의 3-GSPS ADC를 포함하며, 각 경로는 25 dB 범위의 DSA와 최대 1200 MHz 대역폭을 지원하는 DDC를 갖추고 있습니다.1
* **클럭킹**: 내장된 위상 고정 루프(PLL) 및 전압 제어 발진기(VCO)는 외부 기준 클럭으로부터 DAC와 ADC에 필요한 모든 고주파 클럭을 생성하여 시스템 설계를 간소화합니다.2

이러한 통합 DUC/DDC 및 DSA는 단순한 주변 장치가 아니라, 베이스밴드 프로세서(FPGA/ASIC)의 막대한 연산 부담을 덜어주는 필수적인 DSP 블록입니다. 이는 디지털 인터페이스와 전체 시스템 설계를 크게 단순화하는 효과를 가져옵니다.

### 1.3 JESD204B/C 디지털 인터페이스: 데이터 고속도로

AFE7920은 아날로그 프런트엔드와 디지털 베이스밴드 간의 중요한 연결을 위해 8개의 SerDes (Serializer/Deserializer) 트랜시버를 갖추고 있습니다. 이 인터페이스는 JESD204B 및 JESD204C 표준을 모두 준수하며, 최대 29.5 Gbps의 레인 속도로 동작합니다.1 8b/10b 및 64b/66b 인코딩과 다양한 데이터 해상도를 지원하며, 특히 Subclass 1 동기화 기능을 제공하여 결정론적 지연 시간(deterministic latency)을 보장합니다.1

이 고속 인터페이스는 고속 컨버터에서 생성되는 막대한 양의 데이터를 처리하기 위해 필수적입니다. Subclass 1 지원은 여러 채널 또는 여러 AFE7920 장치 간의 위상 관계를 정밀하게 제어해야 하는 빔포밍 및 Massive MIMO와 같은 애플리케이션에서 가장 중요한 기능 중 하나입니다. 이 인터페이스를 올바르게 구성하는 것은 장치를 성공적으로 구동하는 데 있어 가장 어려운 과제 중 하나이며, 바로 이 지점에서 Latte와 같은 도구의 필요성이 대두됩니다.

AFE7920의 아키텍처는 정교한 소프트웨어 추상화 계층의 필요성을 명확히 보여줍니다. 다수의 고속 컨버터, 수많은 통합 DSP 블록(DUC, DDC, DSA, NCO), 복잡한 내부 클럭킹 시스템(PLL/VCO), 그리고 고도로 구성 가능한 29.5 Gbps JESD204C 인터페이스의 조합은 엄청난 수의 구성 레지스터를 생성합니다. AFE79xx 기술 참조 매뉴얼(TRM)은 수백 페이지에 걸쳐 수천 개의 개별 비트 필드를 포함하는 레지스터 맵을 보여줍니다.7 주어진 사용 사례에 대해 이 레지스터들을 수동으로 계산하고 프로그래밍하는 것은 극도로 복잡하고 오류가 발생하기 쉬우며 시간이 많이 소요되는 작업입니다. 이러한 복잡성은 Latte와 같은 도구를 필수적으로 만듭니다. Latte의 목적은 "반송파 주파수 = 3.5 GHz", "대역폭 = 100 MHz"와 같은 높은 수준의 시스템 요구사항을 올바른 저수준 레지스터 쓰기 명령으로 변환하여 하드웨어의 내재된 복잡성을 관리하는 것입니다.

또한, 데이터시트는 "단일 또는 이중 대역 DUC/DDC", "빠른 주파수 스위칭을 위한 이중 NCO", 그리고 빠른 TX/RX 스위칭을 통한 TDD 동작 지원과 같은 기능들을 강조합니다.3 이는 AFE7920이 한 번 설정하고 끝나는 고정 기능 장치가 아니라, 호스트 컨트롤러에 의해 실시간으로 동적으로 재구성되도록 설계되었음을 의미합니다. 이는 시스템 소프트웨어 아키텍처에 중대한 영향을 미칩니다. 제어 소프트웨어는 다양한 동작 모드를 즉석에서 관리하고 적용할 수 있어야 합니다. Latte가 생성하는 구성 파일은 8 초기 전원 인가뿐만 아니라 이러한 동적 동작 모드를 위한 템플릿 역할을 합니다.

| **표 1: AFE7920 주요 성능 사양** |  |
| --- | --- |
| **파라미터** | **사양** |
| 송신(TX) 채널 | 4 |
| 수신(RX) 채널 | 4 |
| 피드백(FB) 채널 | 2 |
| DAC 해상도 / 속도 | 14-bit / 12 GSPS |
| ADC 해상도 / 속도 | 14-bit / 3 GSPS |
| 최대 TX/FB 신호 대역폭 | 1200 MHz |
| 최대 RX 신호 대역폭 | 600 MHz |
| JESD 인터페이스 | 8 레인 @ 29.5 Gbps (JESD204B/C 호환) |
| 패키지 | 17 mm x 17 mm FCBGA (0.8 mm 피치) |
| 데이터 출처: 1 |  |  |

## 섹션 2: Latte 제어 환경: 아키텍처 및 인터페이스

Latte는 AFE7920의 복잡한 하드웨어를 제어하기 위한 기본 소프트웨어 인터페이스입니다. 이 섹션에서는 하드웨어에서 소프트웨어로 초점을 전환하여, Python 기반 도구로서의 Latte의 정체성, GUI 구성 요소, 그리고 그 작동을 지배하는 기본 파일 구조를 탐색합니다.

### 2.1 프레임워크 개요: 단순한 GUI 그 이상

Latte는 AFE77xx 및 AFE79xx 제품군 평가 모듈(EVM)과의 인터페이스를 통해 테스트 및 평가를 수행하기 위한 Python 기반 애플리케이션입니다.8 핵심 기능은 장치를 프로그래밍하는 데 필요한 레지스터 명령들을 생성하여 "구성 파일"이라는 텍스트 형식으로 저장하는 것입니다. 이 파일은 이후 호스트(FPGA 또는 ASIC)에서 AFE를 직접 구동하는 데 사용될 수 있습니다.8 Latte는 하드웨어가 연결되지 않은 상태에서도 "시뮬레이션 모드"로 실행될 수 있어, 오프라인에서 구성 개발을 가능하게 합니다.8

이러한 이중 모드(실시간 EVM 제어 및 오프라인 파일 생성) 기능은 매우 중요한 특징입니다. 이는 RF 시스템 설계와 FPGA/임베디드 소프트웨어 개발을 분리하는 역할을 합니다. RF 엔지니어는 EVM과 Latte를 사용하여 특정 구성을 검증한 후, 해당 레지스터 파일을 생성하여 펌웨어 팀에 전달할 수 있습니다. 이는 매우 효율적이고 병렬적인 워크플로우를 만듭니다.

### 2.2 그래픽 사용자 인터페이스(GUI) 탐색

Latte GUI는 개발의 여러 단계를 지원하기 위해 여러 창으로 구성되어 있습니다.8

* **창 1 (Scripts)**: setup.py, basicBringup.py와 같은 실행 가능한 Python 스크립트 목록을 표시합니다.
* **창 2 (Editor)**: 선택된 스크립트의 코드를 볼 수 있는 구문 강조 기능이 있는 Python 편집기이며, 여기서 직접 코드를 수정할 수 있습니다.
* **창 7 (Command Line)**: 초기 구성 후 LO 주파수나 DSA 설정 변경과 같은 동적 제어를 위해 개별 명령을 입력하고 실행하는 데 사용되는 Python 셸입니다.
* **창 8 (Log)**: 하드웨어로부터의 실시간 상태 메시지, 응답, 경고 및 오류를 표시하는 기본 피드백 메커니즘입니다.
* **Registers View**: 개별 레지스터 값을 실시간으로 확인하고 수정할 수 있는 별도의 인터페이스로, 전문가 수준의 디버깅에 사용됩니다.12

이러한 다중 창 설계는 개발의 각기 다른 단계에 맞춰져 있습니다. Scripts 창은 전체 장치의 초기 구성에, Command Line은 미세 조정 및 대화형 디버깅에, Log 창은 문제 해결에, 그리고 Registers View는 가장 낮은 수준의 접근을 제공합니다.

### 2.3 핵심 스크립트 및 파일 구조

Latte를 설치하면 사용자의 문서 폴더 내에 lib와 projects 폴더가 생성됩니다.8

projects 폴더는 GUI의 Scripts 창에 표시되는 사용자용 스크립트를 포함합니다.8 주요 스크립트는 다음과 같습니다.

* **setup.py**: 연결된 EVM 하드웨어를 확인하고 USB 연결을 설정합니다.
* **devInit.py**: 장치 레지스터 맵과 관련 라이브러리를 로드합니다.
* **basicBringup\_y.py**: 특정 사용 사례에 맞게 장치를 구성하기 위한 파라미터와 함수 호출을 포함하는 핵심 스크립트입니다.8

이 구조를 이해하는 것은 Latte를 사용자 정의하는 데 핵심적입니다. basicBringup 스크립트는 블랙박스가 아닌, 사용자가 편집 가능한 Python 파일입니다. 사용자는 자신의 특정 시스템 요구사항에 맞게 파라미터를 수정하기 위해 대부분의 시간을 이 파일에서 보내게 될 것입니다. 이는 사용자가 미리 정의된 예제를 넘어설 수 있도록 힘을 실어줍니다.

### 2.4 소프트웨어 접근성 및 전제 조건

Latte GUI 설치 프로그램은 TI의 AFE7920 제품 페이지에서 공개적으로 제공되지 않습니다. 사용자는 ti.com의 제품 폴더를 통해 접근을 요청해야 합니다.1 이 요청은 TI의 마케팅 팀에서 검토하며, 접근 권한이 부여되기 전에 비공개 계약(NDA) 체결이 필요할 수 있습니다.13 이는 AFE7920을 사용하는 것이 단순히 도구를 다운로드하는 것만큼 간단하지 않으며, 공식적인 비즈니스 절차가 수반된다는 중요한 물류적 세부사항입니다. 프로젝트 관리자와 엔지니어는 이 필수 소프트웨어에 접근하는 데 필요한 시간을 고려해야 합니다.

Latte는 잠긴 실행 파일이 아닌, 투명한 스크립트 기반 시스템으로 의도적으로 설계되었습니다. TI는 컴파일된 블랙박스 애플리케이션을 제공하는 대신, 핵심 구성 로직(basicBringup.py)이 편집 가능한 텍스트 파일로 노출되는 Python 기반 환경을 제공했습니다.8 이는 모든 고객의 애플리케이션이 고유하며 제한된 옵션을 가진 고정된 GUI로는 불충분하다는 점을 인정한 의도적인 설계 선택입니다. Python 스크립트를 노출함으로써 TI는 사용자가 자신의 구성을 버전 관리하고, 사용자 정의 로직을 추가하며, 구성 프로세스를 자체 개발 환경에 통합할 수 있도록 지원합니다. 이는 사용자를 단순한 운영자가 아닌 개발자로 대우하는 것입니다.

또한, 정적 구성을 위한 "Scripts"와 동적 제어를 위한 "Command Line"의 분리는 라디오의 실제 운영 단계를 반영합니다. basicBringup.py 스크립트는 특정 모드(예: 5G NR Band n78, 100MHz TDD)에 대해 전체 장치를 구성하는 초기화 단계에 직접 매핑됩니다. Command Line은 8 호스트 프로세서가 신호 강도에 따라 디지털 이득(DSA)을 변경하거나 전체 칩을 재초기화하지 않고 주파수를 변경(NCO)하는 것과 같은 작고 동적인 조정을 수행하는 런타임 단계에 매핑됩니다. 따라서 Latte의 UI는 현대 트랜시버에 필요한 제어 아키텍처를 직접적으로 반영한 것입니다.

| **표 2: Latte GUI 구성 요소 기능** |  |  |  |
| --- | --- | --- | --- |
| **구성 요소** | **주요 기능** | **일반적인 사용 사례** |  |
| Scripts 창 | 전체 구성 스크립트 선택 및 실행 | 초기 장치 구동(Bring-up) |  |
| Editor 창 | Python 소스 코드 보기/편집 | 새로운 주파수 계획을 위한 구성 사용자 정의 |  |
| Command Line | 동적 제어를 위한 단일 명령 실행 | 이득 미세 조정 또는 디버깅 |  |
| Log 창 | 하드웨어 응답 및 오류 모니터링 | JESD 링크 문제 해결 |  |
| Registers View | 레지스터 비트 직접 읽기/쓰기 | 전문가 수준의 하드웨어 디버그 |  |
| 데이터 출처: 8 |  |  |  |  |

## 섹션 3: Latte를 통한 AFE7920 구성 워크플로우

이 섹션에서는 Latte를 사용하여 AFE7920을 구성하는 실용적이고 단계적인 가이드를 제공합니다. 이론에서 응용으로 이동하여, 초기 연결부터 최종 구성 파일 생성까지의 전체 프로세스를 상세히 설명합니다.

### 3.1 1단계: 하드웨어 및 소프트웨어 설정

구성을 시작하기 위해서는 적절한 하드웨어 및 소프트웨어 환경을 갖추는 것이 필수적입니다. 최소 8GB, 권장 16GB의 RAM을 갖춘 Windows PC가 필요합니다.8 대상 하드웨어는 AFE7920EVM으로, 이 평가 보드에는 AFE7920 칩, 클럭 생성기(LMK04828), 그리고 TSW14J57EVM과 같은 FPGA 캡처 카드와 인터페이스하기 위한 FMC 커넥터가 포함되어 있습니다.2 프로세스는 Latte GUI에서

setup.py 스크립트를 실행하여 PC와 EVM 간의 USB 연결을 설정하는 것으로 시작됩니다.8

setup.py의 성공적인 실행은 하드웨어가 전원이 켜져 있고, 올바르게 연결되었으며, 소프트웨어에 의해 인식되었음을 확인하는 첫 번째 중요한 이정표입니다.

### 3.2 2단계: basicBringup 스크립트 파라미터화

연결이 확인되면, 사용자는 Editor 창에서 특정 사용 사례에 맞는 basicBringup\_y.py 스크립트를 선택합니다. 구성은 이 스크립트 내의 파라미터를 수정하여 이루어집니다.8 가장 중요한 파라미터 범주는 다음과 같습니다.

* **시스템 파라미터**: sysParams.FRef (기준 주파수), sysParams.Fs (샘플링 속도), 그리고 5개의 내부 PLL에 대한 sysParams.pllLo (LO 주파수)를 포함합니다. sysParams.pllLo를 올바르게 설정하는 것은 RF 반송파를 원하는 주파수에 위치시키는 데 기본이 됩니다.
* **JESD 파라미터**: LMFSHdRx (RX에 대한 JESD 모드), jesdTxLaneMux (ADC 데이터를 SerDes 레인에 매핑), jesdRxLaneMux (DAC 데이터를 SerDes 레인에 매핑) 등이 있습니다. jesdTxLaneMux를 올바르게 구성하는 것은 FPGA가 예상된 레인에서 데이터를 수신하는 데 필수적입니다.

이 단계는 구성 프로세스의 핵심이며, 시스템 사양을 소프트웨어 파라미터로 변환하는 과정입니다.

### 3.3 3단계: 실행 및 구성 파일 생성

스크립트 편집이 완료되면 F5 키를 눌러 스크립트를 실행합니다.8 이 작업은 (EVM이 연결된 경우) 명령을 EVM으로 전송하고, 동시에 텍스트 기반 구성 파일을 생성할 수 있습니다.8 출력 파일 형식은

logDumpInst.logFormat을 통해 설정할 수 있으며, 일반적인 형식은 SPIWrite Addr,Data,lsb,msb입니다.8 생성된 파일은 일반적으로

\Documents\Texas Instruments\Latte\lib 디렉토리에 저장됩니다.8 이 생성된 파일은 AFE7920 구성에 대한 완전한 "레시피"이며, 버전 관리되고, 보관될 수 있으며, 가장 중요하게는 호스트 FPGA의 제어 소프트웨어에 의해 파싱되어 생산 환경의 맞춤형 보드에서 동일한 구성을 정확하게 복제할 수 있습니다.

### 3.4 4단계: 동적 실시간 조정

basicBringup 스크립트가 실행된 후, 사용자는 Command Line (창 7)을 사용하여 대화형으로 장치를 조정할 수 있습니다.8 예를 들어, LO 주파수나 DSA 설정을 즉시 변경할 수 있습니다.8 이를 위해

AFE.PLL[n].configurePll() 및 AFE.TOP.DSA[n].setTxDsa(chNo, dsaSetting)과 같은 특정 함수가 제공됩니다.8 엔지니어는 전체 스크립트를 다시 실행할 필요 없이

setTxDsa 명령을 사용하여 증폭기 이득을 빠르게 스윕하고 스펙트럼 분석기에서 출력을 관찰할 수 있습니다. 이러한 빠른 피드백 루프는 하드웨어 검증 단계에서 매우 중요합니다.

Latte 워크플로우는 표준 엔지니어링 설계 주기를 반영하는 구조화된 프로세스입니다: 정의, 구현, 테스트, 개선. 1단계(설정)는 초기 요구사항 수집 및 환경 설정입니다. 2단계(파라미터화)는 시스템 사양이 코드로 변환되는 핵심 설계 및 구현 단계입니다. 3단계(실행)는 최종 구성을 생성하는 빌드/컴파일 단계에 해당합니다. 4단계(동적 조정)는 대화형 테스트, 디버그 및 개선 단계입니다. 이 구조화된 접근 방식은 AFE 구성이라는 복잡한 작업을 관리 가능하게 만듭니다.

또한, basicBringup.py 스크립트는 RF 하드웨어와 베이스밴드 펌웨어 간의 중요한 "추상화 계약" 역할을 합니다. 스크립트 내의 파라미터(예: JESD 레인 매핑, 데이터 형식)는 FPGA가 준수해야 하는 정확한 프로토콜과 데이터 구조를 정의합니다. RF 엔지니어가 basicBringup.py 스크립트를 확정하고 해당 구성 파일을 생성할 때, 그들은 암묵적으로 FPGA 개발자를 위한 인터페이스 사양을 정의하는 것입니다. 이 파라미터에 대한 모든 변경은 FPGA 로직의 해당 변경을 요구하므로, 스크립트는 하드웨어와 펌웨어 팀 간의 동기화를 위한 중요한 문서가 됩니다.

| **표 3: 핵심 basicBringup.py 시스템 파라미터** |  |  |  |
| --- | --- | --- | --- |
| **파라미터 이름** | **설명** | **데이터 타입** | **예시 값/형식** |
| sysParams.pllLo | TX/RX 채널로 라우팅될 수 있는 PLL0의 반송파 주파수를 설정합니다. | Float (MHz) | 3500.0 |
| sysParams.Fs | RX, TX, FB 경로의 데이터 컨버터 샘플링 속도를 설정합니다. | Float (MHz) | 2949.12 |
| jesdTxLaneMux | 내부 디지털 데이터 스트림과 물리적 SerDes 출력 레인 간의 매핑을 정의합니다. | 정수 리스트 | `` |
| LMFSHdRx | RX 경로에 대한 JESD 모드(예: 8, 4, 2, 1)를 설정합니다. | 정수 | 4 |
| 데이터 출처: 8 |  |  |  |  |

## 섹션 4: 고급 제어 및 자동화 기술

이 섹션에서는 수동 GUI 조작에서 프로그래밍 방식의 자동화로 논의를 확장합니다. Latte의 Python 기반을 활용하여 강력하고 자동화된 테스트 및 특성화 루틴을 만드는 방법을 자세히 설명합니다.

### 4.1 Socket.py 인터페이스: 자동화의 열쇠

Latte의 고급 자동화 기능은 Socket.py 인터페이스를 통해 구현됩니다. Latte는 지정된 포트에서 서버를 생성할 수 있으며 10,

Socket.py (또는 bSocket.py) 스크립트는 이 서버에 연결하는 클라이언트 역할을 합니다.10 이를 통해 외부 Python 프로세스가 Latte의 Command Line에 직접 명령을 입력하는 것처럼 명령을 전송하고 실행할 수 있습니다.10 이 클라이언트-서버 아키텍처는 Latte를 대화형 도구에서 다른 프로그램에 의해 제어될 수 있는 프로그래밍 가능한 엔진, 즉 "헤드리스(headless)" 장비로 변환시키는 관문입니다.

### 4.2 외부 Python에서 자동화된 테스트 스크립트 구축

소켓 연결이 설정되면, 외부 스크립트는 mainWindow.runFile(r"path/to/script.py")와 같은 명령을 전송하여 전체 basicBringup 스크립트를 실행하거나, LO 주파수 또는 DSA 설정을 변경하는 개별 명령을 보낼 수 있습니다.10

Socket.py 스크립트는 더 큰 Python 프로그램에 함수 형태로 통합될 수 있어, 완벽한 프로그래밍 방식의 제어를 가능하게 합니다.10 예를 들어, DSA 설정 목록을 반복하는 Python 루프를 작성하여 각 설정에 대해

setTxDsa 명령을 Latte로 전송하고, 외부 측정 장비에서 측정을 트리거하는 완전 자동화된 특성화 시퀀스(예: 증폭기 이득 압축 스윕)를 구축할 수 있습니다.

### 4.3 측정 장비와의 통합

문서에서는 Socket.py가 스펙트럼 분석기 및 신호 생성기와 같은 측정 장비와 통신하는 데 유용하다고 명시적으로 언급합니다.10 이는 자동화 퍼즐의 마지막 조각입니다. 마스터 Python 스크립트는

Socket.py 함수를 사용하여 Latte를 통해 AFE7920을 제어하는 동시에, PyVISA와 같은 다른 라이브러리를 사용하여 GPIB, USB 또는 이더넷을 통해 테스트 장비를 제어할 수 있습니다. 이를 통해 마스터 스크립트가 테스트 대상 장치(DUT)를 구성하고, 테스트 장비를 구성하며, 측정을 트리거하고, 결과를 기록하는 완전 동기화된 폐쇄 루프 측정 시스템을 사람의 개입 없이 구축할 수 있습니다.

Socket.py 인터페이스는 TI가 LabVIEW, MATLAB 등 모든 가능한 도구에 대한 방대한 드라이버 라이브러리를 제공하고 유지 관리할 필요 없이 시스템 수준의 통합 요구에 대응하는 전략적인 해답입니다. 거의 모든 현대 프로그래밍 및 테스트 환경이 TCP 소켓을 열거나 Python 스크립트를 실행할 수 있다는 사실을 활용하여, 간단하고 보편적인 소켓 기반 텍스트 명령 인터페이스를 제공합니다. 이는 사용자에게 약간의 "접착 코드(glue code)" 작성 부담을 주지만, 그 대가로 엄청난 유연성을 제공하는 우아하고 유지 관리가 적은 방식입니다.

Latte를 중심으로 잘 설계된 자동화 프레임워크는 제품 출시 기간을 극적으로 단축하고 제품 품질을 향상시킬 수 있습니다. AFE7920과 같은 고급 RF 장치를 모든 채널, 주파수 및 모드에 걸쳐 수동으로 특성화하는 것은 엄청난 시간이 소요됩니다. 이 프로세스를 자동화함으로써 팀은 밤새 포괄적인 회귀 테스트를 실행할 수 있습니다. 이는 새로운 구성, 펌웨어 변경 또는 하드웨어 개정에 대한 신속한 검증을 가능하게 하여 개발 주기를 가속화하고 문제가 조기에 발견되도록 보장합니다. 자동화 프레임워크 설정에 대한 초기 투자는 장기적으로 몇 배의 가치로 돌아옵니다.

## 섹션 5: 비교 분석 및 통합 전략

이 섹션에서는 다양한 제어 방법론을 비교하고, Latte를 MATLAB 및 LabVIEW와 같은 일반적인 엔지니어링 워크플로우에 통합하기 위한 구체적인 전략을 제시하며 전략적 조언을 제공합니다.

### 5.1 Latte GUI 대 직접 레지스터 프로그래밍

AFE7920 제어에는 두 가지 주요 접근 방식이 있습니다: Latte GUI를 통한 상위 수준 제어와 SPI를 통한 직접 레지스터 프로그래밍. Latte는 setTxDsa와 같은 높은 수준의 함수 기반 인터페이스를 제공하고, 이를 저수준 레지스터 쓰기 명령으로 변환합니다.8 반면, AFE79xx는 SPI를 통해 직접 제어할 수 있는 광범위하고 복잡한 레지스터 맵을 가지고 있습니다.7

* **Latte 사용**:
  + **장점**: 신속한 개발, 복잡성 감소, 오류 발생 가능성 낮음, TI에서 검증된 구성 사용.
  + **단점**: 추상화로 인해 일부 저수준 세부 정보가 가려질 수 있으며, TI 소프트웨어 환경에 의존적임.
* **직접 레지스터 프로그래밍**:
  + **장점**: 완벽한 제어 가능, 소프트웨어 종속성 없음, 임베디드 시스템에서 잠재적으로 더 빠른 실행.
  + **단점**: 극도로 높은 복잡성, 긴 개발 시간, 잘못된 구성의 높은 위험.

최적의 접근 방식은 이 둘을 결합하는 "하이브리드" 전략입니다. 모든 초기 설계, 평가 및 구동에는 Latte를 사용하고, Latte에서 "골든" 구성 파일을 생성합니다. 그런 다음, 최종 임베디드 애플리케이션에서는 이 사전 검증된 파일을 파싱하고 실행할 수 있는 간단한 SPI 드라이버를 호스트 FPGA/프로세서에 구현합니다. 이 방식은 양쪽의 장점을 모두 취할 수 있습니다.

### 5.2 MATLAB 및 LabVIEW 통합 전략

MATLAB과 LabVIEW는 테스트, 측정 및 데이터 분석을 위한 업계 표준 도구입니다.14 TI에서 AFE7920을 위한 직접적인 MATLAB 또는 LabVIEW 드라이버를 제공한다는 증거는 없습니다. 따라서 통합을 위한 솔루션 아키텍처가 필요합니다. 권장되는 통합 경로는 다음과 같습니다.

1. **MATLAB/LabVIEW (최상위 계층)**: 사용자는 주파수, 전력과 같은 높은 수준의 파라미터를 설정하는 맞춤형 VI 또는 스크립트와 상호 작용합니다.
2. **시스템 호출**: 최상위 스크립트는 마스터 Python 자동화 스크립트를 시스템 명령으로 호출하며, 파라미터를 명령줄 인수로 전달합니다.
3. **Python 스크립트 (중간 계층)**: 이 스크립트는 인수를 수신하고 Socket.py 클라이언트를 사용하여 Latte 서버에 적절한 저수준 명령을 전송합니다.
4. **Latte (실행 엔진)**: Latte는 명령을 수신하고 AFE7920EVM을 프로그래밍합니다.

이 계층화된 접근 방식은 엔지니어가 데이터 수집 및 분석을 위해 선호하는 환경(MATLAB/LabVIEW)에서 작업하면서도, 기본 Latte 제어 엔진의 강력함과 검증된 성능을 활용할 수 있게 해줍니다. 직접적인 MATLAB/LabVIEW 드라이버의 부재는 약점이 아니라, 더 유연하고 강력한 테스트 시스템을 구축할 수 있는 기회입니다. 네이티브 드라이버는 사용자를 특정 소프트웨어 버전과 TI의 구현에 종속시킬 수 있습니다. 반면, Python/소켓 브리지 방식은 더 유연합니다. 이를 통해 테스트 시스템 설계자는 원하는 버전의 MATLAB/LabVIEW를 사용하고, 다른 Python 제어 장비를 원활하게 통합하며, 필요한 경우 제어 로직을 분석 소프트웨어와 다른 머신에서 실행할 수 있습니다. 이는 견고하고 확장 가능한 테스트 시스템의 특징인 모듈식 설계를 강제합니다.

## 섹션 6: 종합 및 전략적 권장 사항

이 마지막 섹션에서는 보고서 전체의 정보를 종합하여 실행 가능한 조언과 모범 사례를 제시하고, 사용자가 일반적인 문제를 해결하고 효과적인 장기 전략을 채택하는 데 도움을 줍니다.

### 6.1 최적의 워크플로우 권장 사항

세 가지 일반적인 시나리오에 대한 권장 워크플로우는 다음과 같습니다.

1. **초기 장치 평가**: AFE7920EVM과 대화형 Latte GUI를 사용합니다. 주요 파라미터의 영향을 이해하는 데 집중합니다.
2. **맞춤형 보드 구동(Bring-Up)**: Latte의 시뮬레이션 모드를 사용하여 기준 구성 파일을 생성합니다. FPGA/MCU에 간단한 SPI 재생 기능을 구현하여 이 파일을 로드합니다. 맞춤형 하드웨어를 디버깅할 때 EVM과 Latte를 "정상 작동이 확인된" 참조로 사용합니다.
3. **자동화된 생산 테스트**: Socket.py 인터페이스를 사용하여 견고한 Python 자동화 프레임워크를 개발합니다. 생산 라인에서 장치를 테스트하기 위해 프로그래밍 방식으로 호출할 수 있는 구성 스크립트 및 테스트 시퀀스 라이브러리를 만듭니다.

### 6.2 일반적인 문제 극복

개발 과정에서 발생할 수 있는 주요 문제점과 해결 방안은 다음과 같습니다.

* **JESD 링크 디버깅**: Latte 구성과 FPGA 로직 간의 JESD 파라미터(레인, 라인 속도, 서브클래스)를 일치시키는 것이 중요합니다. Latte의 Log 창과 FPGA 디버그 도구를 사용하여 링크 상태와 오류를 확인해야 합니다.
* **클럭킹 및 SYSREF**: EVM의 LMK04828은 깨끗한 클럭과 정확한 타이밍의 SYSREF 신호를 생성하는 데 중요한 역할을 합니다. 이는 JESD Subclass 1 동기화에 필수적입니다.
* **Log 창 해석**: Latte의 로그 출력에서 PLL 잠금 해제(unlock), JESD 오류 등 주요 오류 메시지를 식별하고 그 의미를 파악하는 것이 중요합니다.

### 6.3 미래를 대비한 개발

장기적인 성공을 위해, 모든 맞춤형 basicBringup.py 스크립트와 자동화 스크립트를 Git과 같은 버전 관리 시스템을 사용하여 저장소를 구축할 것을 강력히 권장합니다. 이는 재사용 가능하고, 문서화되었으며, 확장 가능한 구성 및 테스트 절차 라이브러리를 생성합니다. 이를 통해 여러 프로젝트와 팀원 간에 일관성을 유지하고 향후 상당한 시간을 절약할 수 있습니다. 제품 수명 주기 전반에 걸쳐 제어 전략은 진화해야 합니다. 평가 단계에서는 대화형 Latte GUI가, 시스템 개발 및 특성화 단계에서는 Python/소켓 자동화가, 최종 생산 펌웨어에서는 Latte 생성 파일을 사용하는 경량 임베디드 드라이버가 가장 효율적이고 견고한 솔루션입니다. 각 프로젝트 단계에 맞는 올바른 도구를 선택하는 것이 성공의 핵심입니다.

#### 참고 자료

1. AFE7920 data sheet, product information and support | TI.com - Texas Instruments, 9월 15, 2025에 액세스, <https://www.ti.com/product/AFE7920>
2. TRF1208-AFE7950-EVM Evaluation Module - „Mouser Electronics“ Lietuva, 9월 15, 2025에 액세스, <https://www.mouser.lt/datasheet/2/405/1/sbau392a-3103440.pdf>
3. AFE79xx Quad-Channel RF Transceiver With ... - Texas Instruments, 9월 15, 2025에 액세스, <https://www.ti.com/lit/gpn/AFE7920>
4. AFE77xxD Latte series | TI.com - Texas Instruments, 9월 15, 2025에 액세스, <https://www.ti.com/video/series/AFE77xxD-latte-series.html>
5. Texas Instruments AFE7920 | Demoboard AFE7920EVM | Würth Elektronik Reference Design Search, 9월 15, 2025에 액세스, <https://www.we-online.com/en/components/icref/texas-instruments/AFE7920-AFE7920EVM-RF-signals-communications>
6. AFE7920EVM Texas Instruments - Mouser Electronics, 9월 15, 2025에 액세스, [https://www.mouser.com/ProductDetail/Texas-Instruments/AFE7920EVM?qs=xZ%2FP%252Ba9zWqZKLv0yhXygJg%3D%3D](https://www.mouser.com/ProductDetail/Texas-Instruments/AFE7920EVM?qs=xZ/P%252Ba9zWqZKLv0yhXygJg%3D%3D)
7. AFE79xx Programming User Guide - Technical Reference ... - TI E2E, 9월 15, 2025에 액세스, <http://e2e.ti.com/cfs-file/__key/communityserver-discussions-components-files/73/TRM.pdf>
8. AFE77xx Latte GUI - Texas Instruments, 9월 15, 2025에 액세스, <https://www.ti.com/lit/pdf/sbau340>
9. AFE7920EVM Evaluation board | TI.com - Texas Instruments, 9월 15, 2025에 액세스, <https://www.ti.com/tool/AFE7920EVM>
10. Controlling Latte GUI from Python - Texas Instruments, 9월 15, 2025에 액세스, <https://www.ti.com/lit/pdf/sbaa397>
11. Afe77xx Latte Gui | PDF | Graphical User Interfaces | Computer File - Scribd, 9월 15, 2025에 액세스, <https://www.scribd.com/document/900822427/Afe77xx-Latte-Gui>
12. TI-Latte Training for the OPT3101EVM | Video | TI.com - Texas Instruments, 9월 15, 2025에 액세스, <https://www.ti.com/video/5837161210001>
13. AFE7920: how can i get latte gui for setting afe 7920 - RF & microwave forum - TI E2E, 9월 15, 2025에 액세스, <https://e2e.ti.com/support/rf-microwave-group/rf-microwave/f/rf-microwave-forum/1328855/afe7920-how-can-i-get-latte-gui-for-setting-afe-7920>
14. LabVIEW - Purpose-built development environment for automated test and measurement - Third-Party Products & Services - MATLAB & Simulink - MathWorks, 9월 15, 2025에 액세스, <https://www.mathworks.com/products/connections/product_detail/labview.html>
15. LabView or Matlab? | Automation & Control Engineering Forum, 9월 15, 2025에 액세스, <https://control.com/forums/threads/labview-or-matlab.23327/>
16. Can anyone explain the difference between labview and matlab/simulink? - Reddit, 9월 15, 2025에 액세스, <https://www.reddit.com/r/AskEngineers/comments/35brt4/can_anyone_explain_the_difference_between_labview/>