Hardware Deployment Guide

Version: 1.0
Date: 2025
Project: POC for Real-Time Deployment of ECGFounder Foundation Model on Edge Hardware

Abstract

This guide describes the hardware deployment and system configuration for real-time 12-lead ECG interpretation using the ECGFounder foundation model. The deployment achieves 115 ms end-to-end latency on ARM Cortex-A72 processors, enabling clinical decision support in resource-constrained settings. All performance metrics and configuration parameters are based on empirical validation conducted as part of this research project.

1. System Architecture

End-to-end pipeline:

```
ECG Acquisition Device (12-lead, 500 Hz)
    ↓ [Digital signal interface]
Edge Computer (ARM or x86-64)
    ├── Signal Quality Control
    ├── Preprocessing Pipeline (14.1 ± 1.6 ms on ARM)
    ├── ONNX Runtime Inference (100.9 ± 4.8 ms on ARM)
    └── Post-processing
    ↓
Diagnostic Output : JSON
```

Total latency: 115.0 ± 5.2 ms on ARM Cortex-A72 (validated on Raspberry Pi 4)

2. Hardware Requirements

2.1 Recommended Platform: Raspberry Pi 4 Model B

Specifications:

- **CPU:** ARM Cortex-A72 @ 1.5 GHz (4 cores, 64-bit ARMv8-A)
- **RAM:** 4 GB LPDDR4-3200 (minimum 2 GB for inference-only deployment)
- **Storage:** 32 GB microSD card (Class 10 or higher)
- **Power:** 5V/3A USB-C (15W)
- **Cooling:** Passive heatsink recommended for sustained operation

Rationale: The ARM Cortex-A72 provides real-time performance (<500 ms medical device standard) at significantly lower cost and power consumption compared to x86-64 or GPU-based systems, making it suitable for point-of-care deployment in resource-limited settings.

2.2 Alternative Platform: x86-64

Minimum specifications:

- **CPU:** Intel Core i5 (6th gen or newer) or AMD Ryzen 5
- **RAM:** 4 GB DDR4
- **Storage:** 10 GB free space

Performance: 57.50 ± 2.48 ms total latency (2.0× faster than ARM)

Use cases: Development, testing, batch processing of archived ECG databases

2.3 Hardware Comparison

Platform	CPU	Latency (ms)	Hardware Cost	Efficiency*
ARM (Raspberry Pi 4)	Cortex-A72 @ 1.5 GHz	115.0 ± 5.2	Low	**10.5**
x86-64 (Desktop)	Core i7 @ 3.8 GHz	57.5 ± 2.5	Moderate	0.44

*Efficiency = (AUROC × 1000) / (Latency), higher is better

Note: Hardware costs vary by region and time. ARM platforms typically offer 5-10× lower cost than comparable x86-64 systems with sufficient performance for this application.

3. Software Environment Setup

3.1 Operating System

Raspberry Pi 4:

- OS: Raspberry Pi OS (64-bit, Debian 11 "Bullseye")
- Installation: Use Raspberry Pi Imager (https://www.raspberrypi.com/software/)
- Configuration: Enable SSH, configure network

x86-64:

- OS: Ubuntu 20.04 LTS or 22.04 LTS
- Kernel: Linux 5.4+ (for optimal ONNX Runtime performance)

3.2 System Dependencies

Update package manager:

```
sudo apt update &amp;&amp; sudo apt upgrade -y
```

Install dependencies:

```
sudo apt install -y python3-pip python3-venv git wget \
    build-essential libatlas-base-dev libhdf5-dev
```

Rationale:

- `build-essential`: Required for NumPy compilation on ARM
- `libatlas-base-dev`: BLAS/LAPACK for linear algebra operations
- `libhdf5-dev`: HDF5 support for WFDB file format

3.3 Python Environment

Create virtual environment:

```
python3 -m venv venv-ecg
source venv-ecg/bin/activate
pip install --upgrade pip setuptools wheel
```

Install dependencies (requirements.txt):

```
numpy==1.21.6
scipy==1.7.3
onnxruntime==1.16.3
wfdb==4.1.2
pandas==1.3.5
```

Installation:

```
pip install -r requirements.txt
```

Installation time: Approximately 15-20 minutes on Raspberry Pi 4 (NumPy compilation from source)

4. Model Conversion & Validation

4.1 PyTorch to ONNX Conversion

Conversion script: `src/inference/pytorch_to_onnx_converter.py`

Command:

```
python src/inference/pytorch_to_onnx_converter.py \
    --checkpoint models/checkpoint/12_lead_ECGFounder.pth \
    --output models/onnx/ecg_founder_150class.onnx \
    --opset 14
```

Validation: Numerical equivalence verified via paired inference on 100 random samples:

- Pearson correlation: $r = 0.9995$ ($p < 0.001$)
- Max absolute error: 1.23×10^{-6}

Validation script: `src/inference/paired_eval_pt_vs_onnx.py`

Model size: 112 MB (FP32 precision)

5. System Configuration

5.1 CPU Frequency Scaling

Objective: Maintain maximum CPU frequency for consistent latency

Check current governor:

```
cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

Set performance mode:

```
echo performance | sudo tee /sys/devices/system/cpu/cpu*/cpufreq/scaling_governor
```

Expected improvement: 5-10% reduction in latency variance

5.2 ONNX Runtime Configuration

Environment variables (add to ~/.bashrc):

```
export OMP_NUM_THREADS=4  # Match CPU core count
export ORT_ENABLE_EXTENDED=1  # Enable graph optimizations
```

Rationale: Thread count matching prevents CPU oversubscription; extended optimizations enable kernel fusion and constant folding.

6. Performance Validation

6.1 Latency Benchmarking

Benchmark script: `src/validation/benchmark_ort_arm64.py`

Command:

```
python src/validation/benchmark_ort_arm64.py \
    --model models/onnx/ecg_founder_150class.onnx \
    --samples 10 \
    --iterations 200
```

Expected results (Raspberry Pi 4):

Component	Mean (ms)	Std Dev (ms)
Preprocessing	14.1	1.6
ONNX Inference	100.9	4.8
Total	**115.0**	**5.2**

Real-time criterion: Medical device standard IEC 60601-2-47 specifies <500 ms latency for ambulatory ECG systems. This deployment meets the requirement with 4.3× safety margin.

6.2 Diagnostic Accuracy Validation

Validation script: `src/validation/ecgfounder_post_training_complete.py`

PTB-XL test set (fold 10, n=2,163):

- Macro AUROC: 0.909 (95% CI: [0.906, 0.912])
- Per-class AUROC: Available in `results/validation/metrics_test_final-last.json`

7. ECG Device Integration

7.1 Input Signal Requirements

Format specifications:

- **Channels:** 12 leads (I, II, III, aVR, aVL, aVF, V1-V6)
- **Sampling rate:** 500 Hz (required for ECGFounder preprocessing)
- **Duration:** 10 seconds (5,000 samples per lead)
- **Amplitude:** Millivolts (mV)
- **Data type:** Float32 or Int16 (with appropriate gain conversion)

Important: System processes **raw digital ECG signals**, not scanned paper ECG images. Image-based digitization introduces prohibitive accuracy degradation.

7.2 Supported File Formats

WFDB (PhysioNet):

```
import wfdb
record = wfdb.rdsamp('path/to/ecg_file')
signal = record.p_signal.T  # Shape: (12, 5000)
```

CSV/NumPy:

```
import numpy as np
signal = np.loadtxt('ecg.csv', delimiter=',')  # Shape: (12, 5000)
```

DICOM Waveform: Parse via `pydicom` library (DICOM Supplement 30)

HL7 aECG: Parse via XML parser (HL7 CDA Release 2.0)

7.3 Signal Quality Control

Pre-inference quality checks (implemented in `src/preprocessing/`):

1. **Saturation detection:** >5% samples at ADC limits → reject
2. **Flatline detection:** Variance <0.01 mV^2 in any 1-second window → reject
3. **Lead-off detection:** RMS amplitude <0.05 mV → reject

Quality score: QS = 1 - (saturated + flatline) / total_samples

Acceptance threshold: QS > 0.95

8. Troubleshooting

8.1 Slow Inference (>200 ms on Raspberry Pi)

Diagnosis:

```
# Check CPU throttling
vcgencmd get_throttled
# 0x0 = No throttling
# 0x50000 = Currently throttled

# Check temperature
vcgencmd measure_temp
# Target: &lt;65°C
```

Solutions:

1. **Thermal:** Install heatsink or cooling fan if temperature >70°C
2. **Power:** Verify 5V/3A power supply (measure with `vcgencmd measure_volts`)
3. **Governor:** Force performance mode (see Section 5.1)

8.2 ONNX Runtime Errors

Error: "Failed to load model"

- Verify ONNX opset version matches runtime (opset 14 required for ONNX Runtime 1.16.3)
- Re-export model if necessary

Error: "Memory allocation failed"

- Check available RAM: `free -h` (minimum 500 MB free required)
- Close background processes or enable swap memory

8.3 Incorrect Predictions

Common issues:

1. **Lead order mismatch:** Verify device lead sequence matches expected order (I, II, III, aVR, aVL, aVF, V1-V6)
2. **Incorrect voltage gain:** Check ADC conversion factor. Expected Lead II amplitude: 0.5-2.0 mV for typical QRS complex
3. **Sampling rate mismatch:** Resample to 500 Hz if device uses different rate:

```
from scipy.signal import resample
ecg_resampled = resample(ecg_signal, 5000, axis=1)
```

9. Security Considerations

9.1 Data Privacy

HIPAA/GDPR compliance:

- Never log identifiable patient information (PHI)
- Use cryptographic hashes for patient IDs in audit logs
- Encrypt data at rest and in transit (TLS 1.3 for network communication)

9.2 System Hardening

Disable unnecessary services:

```
sudo systemctl disable bluetooth
sudo systemctl disable avahi-daemon
```

Enable firewall:

```
sudo apt install ufw
sudo ufw default deny incoming
sudo ufw allow ssh
sudo ufw enable
```

Automatic security updates:

```
sudo apt install unattended-upgrades
sudo dpkg-reconfigure unattended-upgrades
```

10. Performance Optimization

10.1 Model Quantization (Optional)

INT8 quantization (dynamic):

```
from onnxruntime.quantization import quantize_dynamic, QuantType

quantize_dynamic(
    model_input='ecg_founder_150class.onnx',
    model_output='ecg_founder_150class_int8.onnx',
    weight_type=QuantType.QInt8
)
```

Expected results:

- Model size reduction: 50-75%
- Latency reduction: 30-50%
- Accuracy impact: <1% AUROC drop (requires validation)

Warning: Quantization must be validated on PTB-XL test set before clinical deployment.

10.2 Batch Processing

For offline analysis:

- Batch size 10: ~2.5× throughput improvement (450 ms for 10 ECGs)
- Trade-off: Higher total latency per sample

11. References

[1] Li, J., et al. (2024). "An Electrocardiogram Foundation Model Built on over 10 Million Recordings with External Evaluation across Multiple Domains." NEJM AI, 1(7).

[2] Wagner, P., et al. (2020). "PTB-XL, a large publicly available electrocardiography dataset." Scientific Data, 7(1), 154.

[3] IEC 60601-2-47:2012. "Medical electrical equipment - Part 2-47: Particular requirements for the basic safety and essential performance of ambulatory electrocardiographic systems."

[4] ONNX Runtime Documentation. (2024). "Performance Tuning." https://onnxruntime.ai/docs/performance/

[5] Raspberry Pi Foundation. (2023). "Raspberry Pi 4 Model B Documentation." https://www.raspberrypi.com/documentation/

Validated Platforms: Raspberry Pi 4 (4GB RAM), Ubuntu 22.04 x86-64

Contact: dr.ghasemdolatkhah@gmail.com | https://www.linkedin.com/in/ghasemdolatkhah-md | https://github.com/GhIrani33/ecgfounder-edge-cds

Last Updated:2025