T-CHIP GROUP

Firefly RK1828
Edge LLM Solution

T-firefly.com

Overview

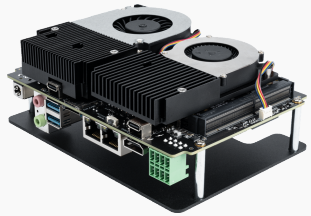
RK1828 Multi-Card Compute Pool

Edge AI devices are typically constrained by compute and memory, and can only run small models. The Firefly RK1828 multi-card pooling solution pairs an RK3588 host with multiple RK1828 NPU accelerator cards and runs pooled, collaborative inference across them. LLM, vision, audio, and agent applications all run locally, with scalable configurations: up to 7B on one card, up to 12B on two cards, and up to 27B on four cards.



Product Overview





AIBOX PRO / AIBOX PRO KIT

Dimensions
Computer: 160.0 × 111.0 × 61.0 mm
Dev Kit: 148.0 × 106.0 × 36.4 mm

Compute Config
RK3588 + 2 × RK1828

Large Model Support
Qwen3.5-9B

Interfaces
Dual GbE, CAN-FD, RS485, Isolated DI/DO, HDMI 2.1, etc.





AIBOX QUA / AIBOX QUA KIT

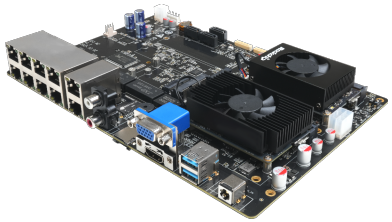
Dimensions
Computer: 128.0 × 128.0 × 49.5 mm
Dev Kit: 126.2 × 120.0 × 29.0 mm

Compute Config
RK3588 + 4 × RK1828

Large Model Support
Qwen3.8-27B

Interfaces
5GbE, 1GbE, M.2, USB 3.0, HDMI 2.1, TF card, etc.

Product Overview



RK1828 Dev Kit

Dimensions
232.97 × 162.8 × 34.86 mm

Compute Config
1 × RK3588 + 1 × RK1828

Product Form
Dev Kit (Dual 260P Module)

Interfaces
Dual Gigabit Ethernet, USB 3.0, HDMI, Mini PCIe, RS485, CAN-FD, Isolated DI/DO, etc.



GS1-N2 AI NVR

Dimensions
490.0 × 368.0 × 44.4 mm

Compute Config
1 × RK3588 + 1 × RK1828

Product Form
1U Rack-Mounted Server

Interfaces
POE, 1G Ethernet, SATA 3.0, Isolated DI/DO, VGA, etc.



CSB2-N10

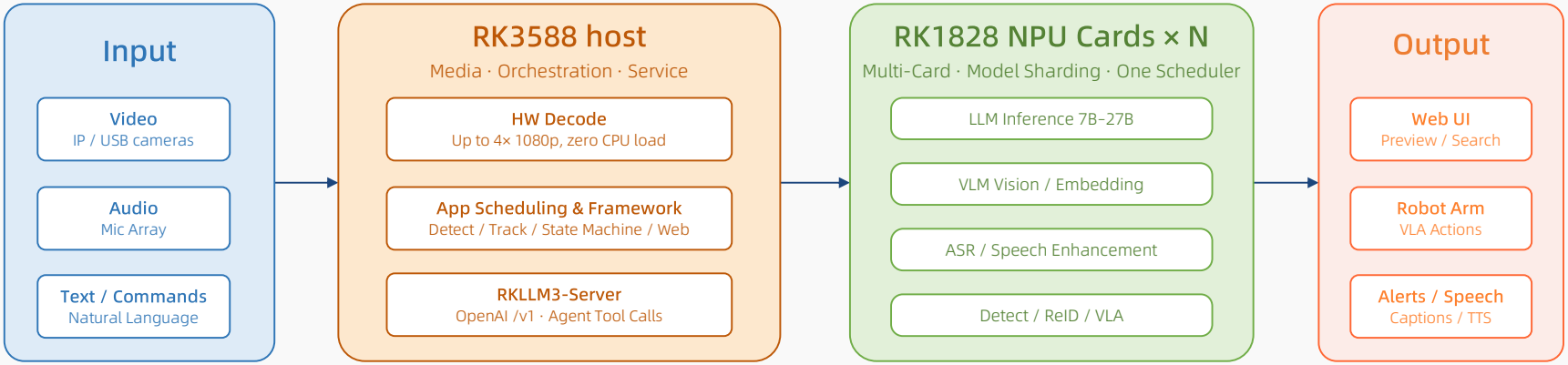
Dimensions
487.9 × 530.59 × 88.8 mm

Compute Config
5 × RK3588 + 4 × RK1828

Product Form
2U Rack-Mounted Server

Interfaces
6 x SATA 3.0, VGA, Console, Dual 10 Gigabit SFP+, Dual Gigabit Ethernet, etc.

RK3588 + N × RK1828



01 Elastic Compute Scaling

PCIe multi-card cascade under one scheduler: up to 7B on one card, 12B on two, 27B on four. The model is partitioned across cards and runs as a pipeline.

02 Low Host Load

All inference runs on RK1828. The RK3588 CPU stays at only 2.4–2.6% and is free for app logic and services.

03 Decode and Inference in Parallel

MPP hardware decode and NPU inference are isolated: decoding adds no CPU load, and inference never blocks the preview.

04 Low Power on Device

A 2-card system averages 30.5 W and a 4-card system 57.8 W – no GPU-server power, size, or cooling burden. Built for embedded use.

05 Local Data Loop

LLM, VLM, and ASR all run on-device. Audio and video stay on the device, and everything keeps working offline.

06 Plug-and-Play Ecosystem

OpenAI /v1 compatible, so OpenClaw, Nanobot, and other agent frameworks connect directly. Register an MCP tool to add device skills.

Legend In/Out CPU NPU UI

Performance

Multi-Card LLM

RK3588 host · RK1828 SO-DIMM · W4A16 quantized · measured with the cards on PCIe

Qwen3.5-9B

2 × RK1828 · 4K-token in / 128-token out

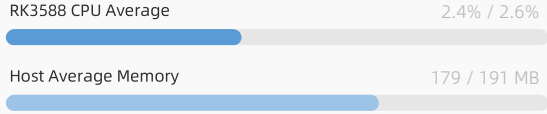
4.80s	853.7	33.19
TTFT	Prefill TPS	Decode TPS

Qwen3.8-27B

4 × RK1828 · 4K-token in / 128-token out

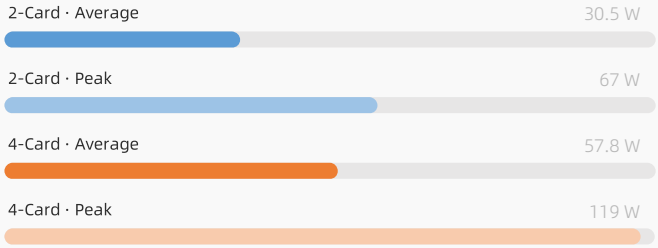
6.92s	592.1	13.02
TTFT	Prefill TPS	Decode TPS

Very low host overhead



During the 9B (2-card) and 27B (4-card) runs the RK3588 stays at about 2.5% CPU, leaving room for apps and decode.

Low System Power

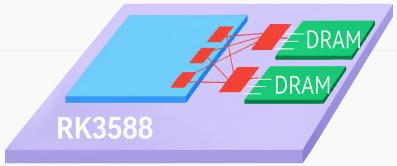


A 27B run draws about as much as a laptop – no server room needed.

One architecture, many on-device use cases

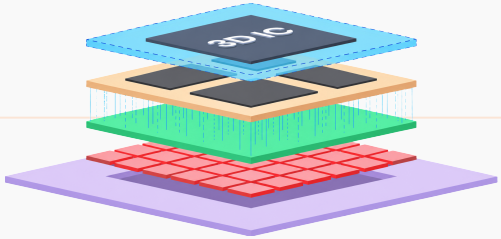
- **Video Event Search** — Multi-cam preview + smart alerts
- **Traffic / Inspection** — Multi-channel detection + AI SOP
- **Multimodal Search** — Person tracks / archive
- **AI Meeting / Avatar** — Transcription & notes / KB Q&A
- **Robotics / Smart Home** — VLA picking / offline voice
- **Edge Agent / Local LLM** — ClawChips / local LLM

RK1828: 3D DRAM



Conventional 2D Chip

Structure	Dies sit side by side on the package
DRAM	Few DRAM lanes (32–128-bit bus)
Bandwidth	Long wires, higher R and C: higher latency, less bandwidth
Efficiency	Driving long wires costs more power and lowers efficiency



3D Stacked Chip

Structure	Dies are stacked vertically for a 3D architecture
DRAM	Wide DRAM interface (>10,000 data lanes)
Bandwidth	Very short vertical links: much lower latency and power, very high bandwidth
Efficiency	Short links cut drive power and raise efficiency

Use Case 1: Video Event Analysis & Search

Overview

A local multi-channel vision system on RK3588. It hardware-decodes up to 4 video streams with no host CPU load, so several cameras preview together smoothly. A VLM reads the scene and writes a plain-language alert, so every alert is easy to understand.

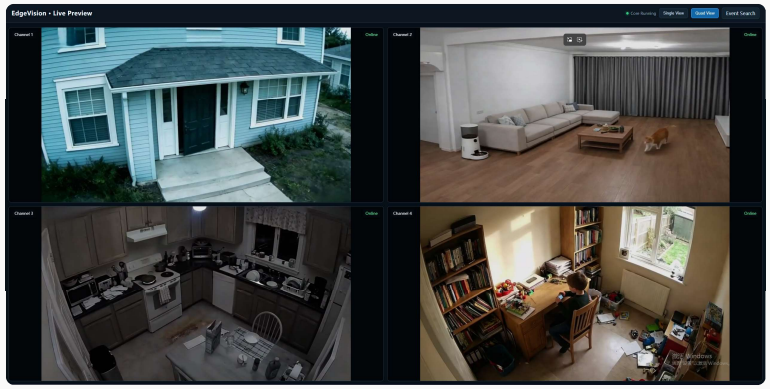
Search

Events are stored as vectors and can be searched in plain language: with two RK1828 cards, one sentence finds the matching past clip. With a single card you can still browse event history – it degrades gracefully.

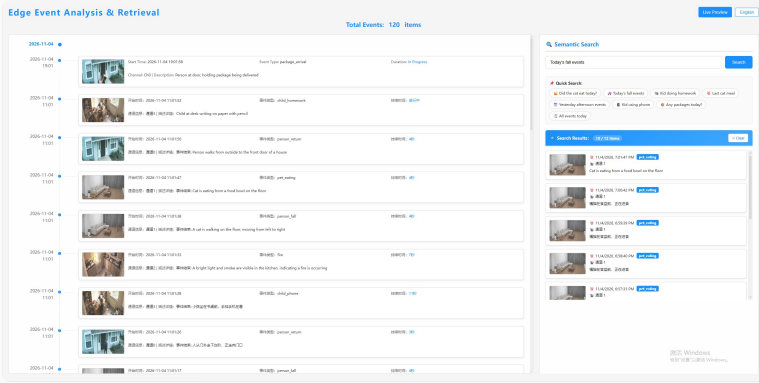
Use Cases

It runs entirely on the local network, with no cloud dependency. Ideal for home monitoring with 4 cameras (door, living room, kitchen, kids' room), retail stores and campus security, and review of past inspection events.

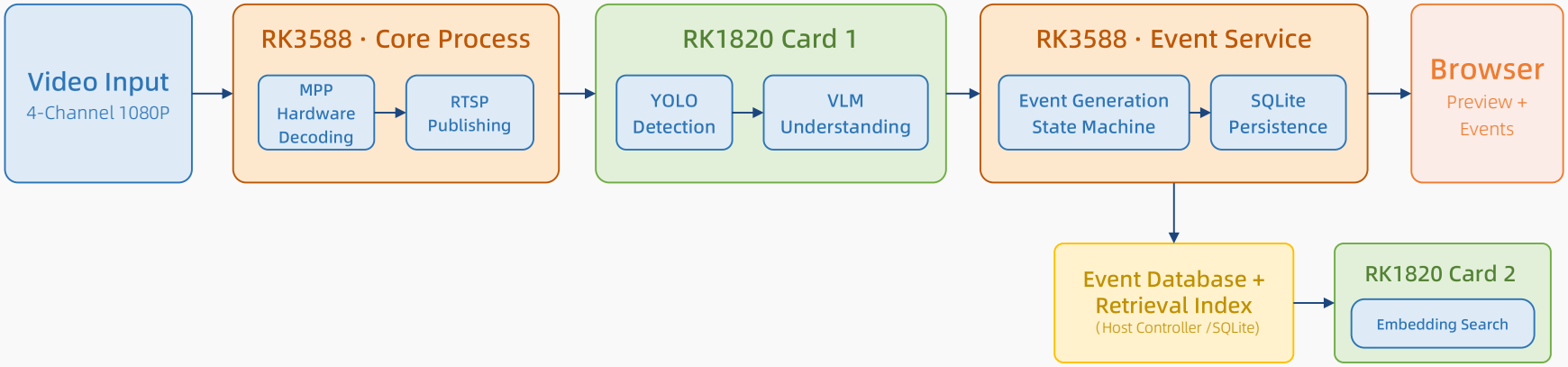
- Live Multi-View Preview



- On-Device Event Analysis & Search



Use Case 1: Video Event Analysis & Search



Performance:

- 4× 1080p hardware decode via MPP, no CPU load;
- One card already gives live preview + event detection;
- Card 2 adds semantic event search; without it the core still works;
- Low-latency preview in the browser via WebRTC.

Architecture:

- Three processes: C++ core + MediaMTX + Python web (over Unix domain sockets);
- Event = VLM understanding + state machine (detected / ongoing), written to SQLite (auto event_id);
- Preview, detection and history browsing are independent;
- The multistream_vlm family reuses the same core.

Legend In/Out CPU NPU Agent Data UI

Use Case 2: Multimodal Analysis & Search

Overview

A local multi-camera person search system on RK3588 + RK1828 using pure ReID: sample each video every 0.5 s, detect people with YOLOv8 plus geometric filtering and NMS, extract 512-D appearance features with OSNet into time-stamped tracks, merge tracks into one person per video by greedy cosine clustering (threshold 0.85), then match across videos to link the same person on other cameras and at other times.

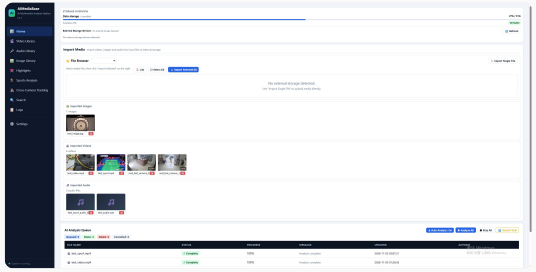
Search

The UI shows a track timeline, a cross-camera journey (time cards in a row), find-this-person (Top-K gallery with similarity badges) and colored ID boxes on the player. A debug view records every YOLO box and the reason it was discarded, so you can see where a miss happened.

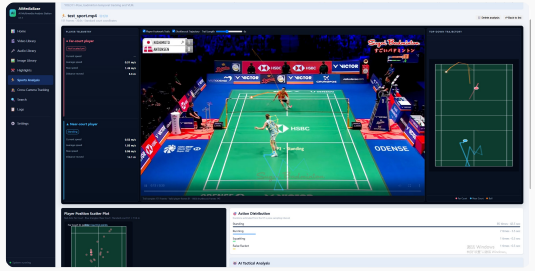
Use Cases

Ideal for multi-camera search in parks, residential compounds, and malls (finding a missing person, retracing a route), bulk footage analysis, and movement analysis.

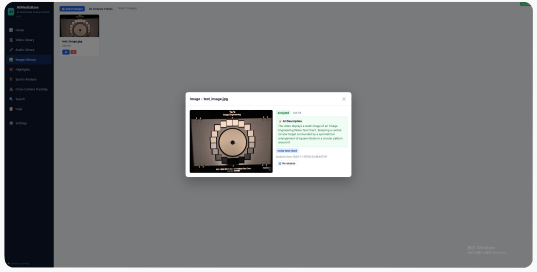
- Analysis Dashboard



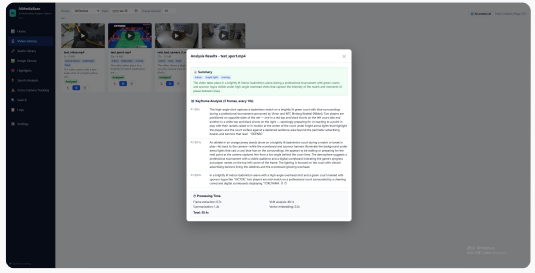
- Video Analysis



- Image Analysis



- Audio Analysis



Use Case 2: Multimodal Analysis & Search



Performance:

- All six models run on RK1828 (VLM / Embedding / ASR / Pose / YOLO / OSNet);
- OSNet 512-D at W8A16; cosine 0.85 in-video, 0.60 cross-video;
- 0.5 s sampling interval; greedy tracklet-to-person merge;
- Pure ReID never calls the VLM; only the CPU handles CIF for ASR.

Architecture:

- FastAPI and SQLite handle orchestration; GStreamer handles media;
- One Paraformer subprocess per WAV, so failures stay isolated;
- Vectors plus captions enable plain-language search;
- Across cameras: detect, filter, NMS, crop, feature, cluster, match.

Legend In/Out CPU NPU Agent Data UI

Use Case 3: Traffic Monitoring

Overview

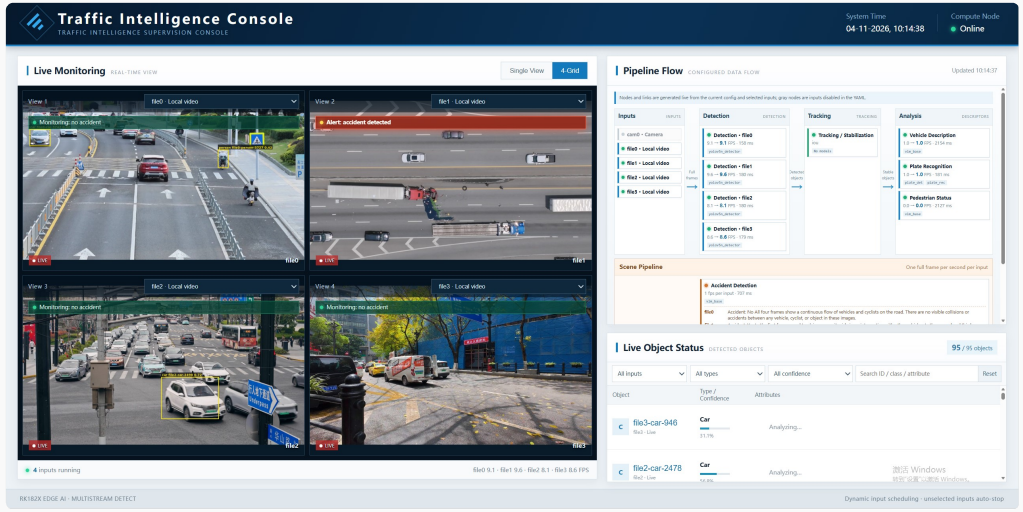
Multi-channel traffic video analysis on an RK1828 Debian board: GStreamer ingests the streams, each running its own RKNN3 pipeline where YOLOv5n detects vehicles and people (up to 20 FPS per channel), followed by IoU tracking and ID stabilization. Plate detection + PP-OCR and VLM captions of people and vehicles feed downstream queues, and results appear as OSD boxes, media streams, and HTTP/SSE data.

Benefits

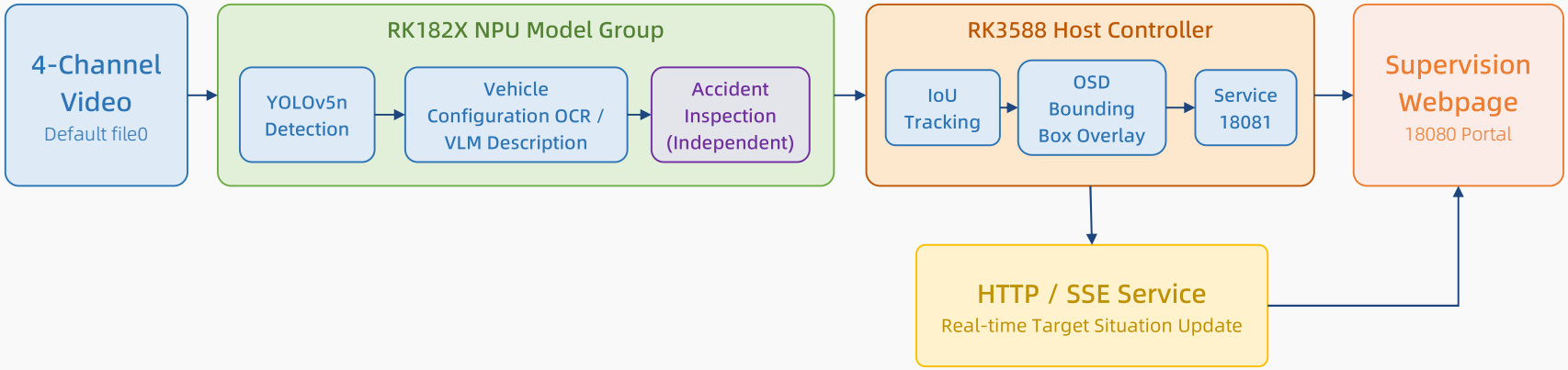
It also runs full-frame accident checks: one full frame per channel per second, and after four frames it calls the accident LLM once to state in plain text whether the scene looks like an accident. The right panel shows live FPS per node and the verdict.

Use Cases

Ideal for intersection and checkpoint monitoring (vehicles, people, plates), accident review, campus roads, and car parks.



Use Case 3: Traffic Monitoring



Performance:

- Up to 20 FPS detection per channel (YOLOv5n; Pipeline limit);
- 8-instance pool for detection; the VLM uses the remaining 4 NPU cores and shares 1 instance;
- The accident check batches 4 frames per LLM call, sharply cutting the number of calls;
- Up to 4 channels at once; unused inputs are switched off to save compute.

Architecture:

- Generic framework: input, queues, model pool, scheduler, object store, media out;
- New scenario = scenario YAML + typed callback + scenario page;
- IoU tracking keeps stable object IDs; OSD overlays boxes and attributes;
- Single or quad view; records are retained for 3 minutes and can be filtered.

Legend In/Out CPU NPU Agent Data UI

Use Case 4: AI Meeting Assistant

Overview

A local meeting audio system on RK3588 + RK1828 covering recording, live transcription and automatic notes: RockASR does speech recognition, RockTextSum writes the summary into summary/, RockRAG answers questions about it, and RockVQE cleans the audio - one chain from capture to transcript to summary to Q&A.
Unlike cloud-based meeting products, all inference runs on the device, and recordings never leave it. A Vue page talks to the main program over WebSocket, so text appears live during the meeting and the notes are ready when it ends.

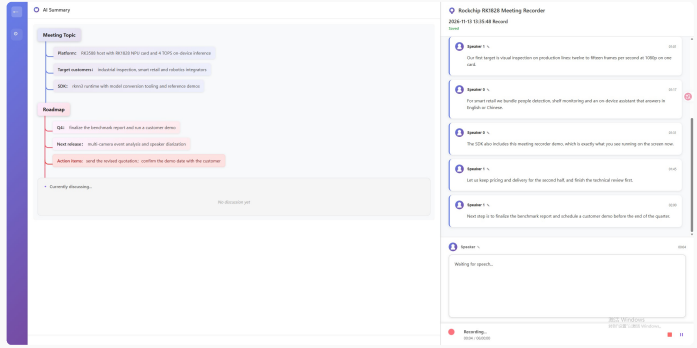
Access

Everything is accessed from a browser: open http://localhost:3000 on the device, or http://<board IP>:3000 from a PC on the same LAN. The page connects to ws://<board IP>:8081, so allow ports 3000 (Web UI) and 8081 (WebSocket) in the firewall.

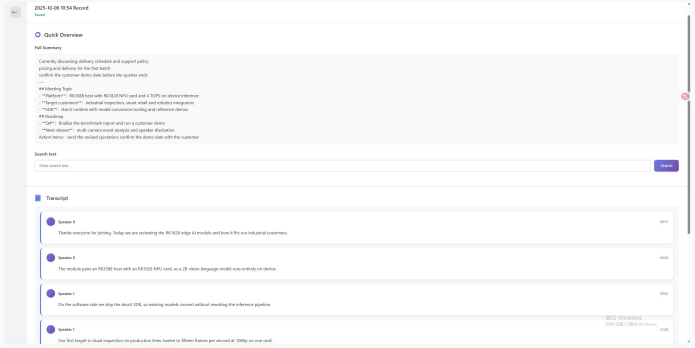
Use Cases

Ideal for meeting-room records, interview and training archives, and Q&A over past company meetings.

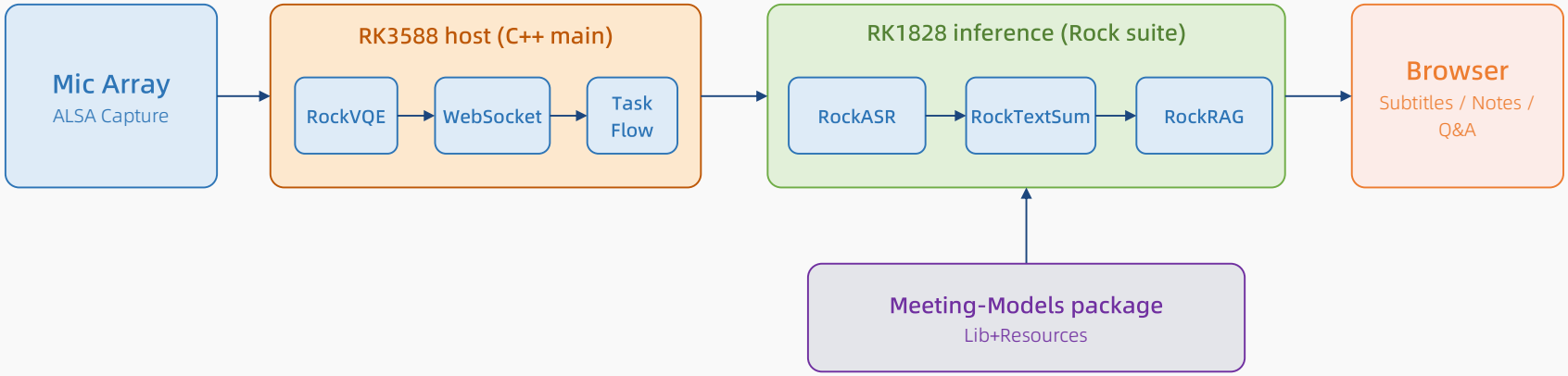
- Live Transcription



- Smart Meeting Summary



Use Case 4: AI Meeting Assistant



Performance:

- Fully offline: ASR / summary / RAG all on RK1828, no cloud;
- Set the CPU/devfreq governor to performance for faster inference;
- Streaming recognition gives live subtitles;
- Single C++ process, low overhead.

Architecture:

- Audio chain: ALSA + RockVQE (echo cancellation / noise suppression / AGC);
- Rock meeting suite: RockASR / RockTextSum / RockRAG must be version-matched;
- WebSocket pushes subtitles, notes and answers;
- Supports speaker diarization and more.

Legend In/Out CPU NPU Agent Data UI

Use Case 5: ClawChips Agent App

Overview

ClawChips uses its own local routing to keep interaction costs low. When a request arrives it first reuses past model choices from memory; if nothing matches, it decides whether to run cloud or local, then saves that decision so it keeps improving – cutting wasted cloud calls at the source. On PinchBench it cut cloud model usage by 40%: local inference at zero cost, cloud only when it helps.

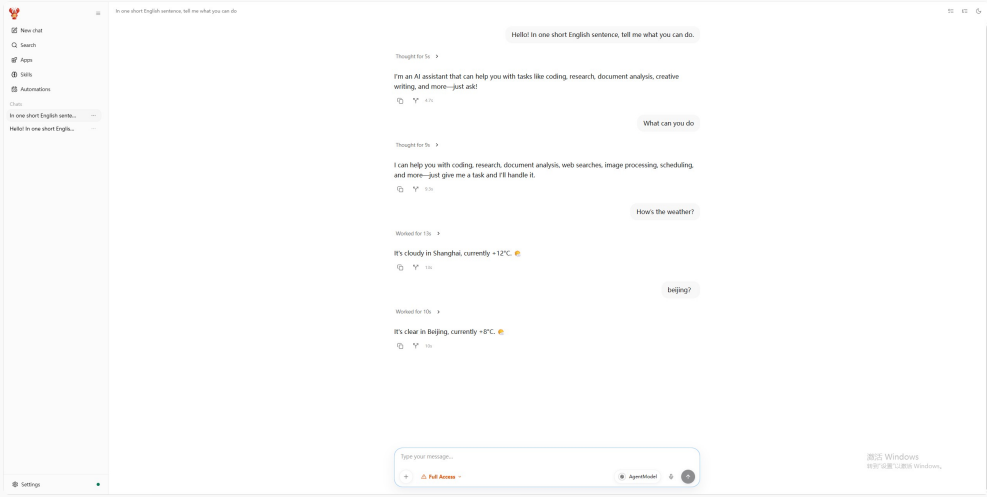
Model

ClawChips has also heavily tuned a 4B on-device model with RKNN3 SDK quantization, so it runs efficiently on RK1828 with minimal accuracy loss – steady compute for the agent's language understanding and task decisions.

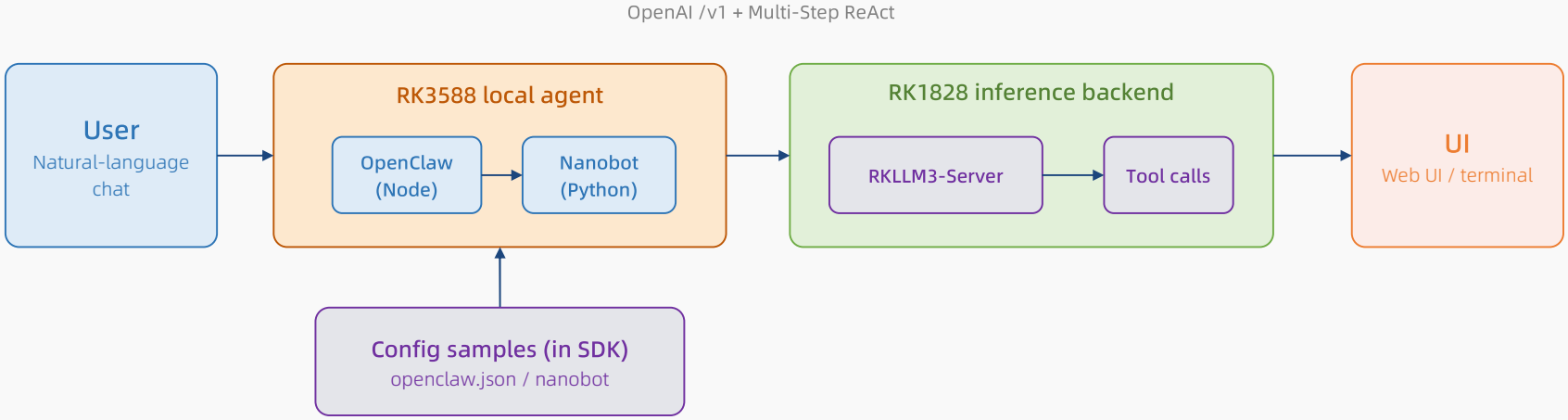
Use Cases

Ideal for on-device AI assistants, agent pilots on private networks or offline, and plugging a local LLM into existing workflows.

- ClawChips UI



Use Case 5: ClawChips Agent App



Performance:

- Fully local: agent framework + 4B model on one device, no cloud API, and no data leaves the device;
- Local AgentModel V3.1 4B (shared package for App Portal and Llama-RKNN3-Server);
- OpenAI-compatible API, so any custom Provider client can connect;
- Two ecosystems: OpenClaw (Node 22) / Nanobot (Python)

Architecture:

- OpenClaw: web UI on 18789, skill orchestration, session management
- Nanobot: on-device setup; Providers.custom points to the local port
- RKLLM3-Server serves /v1 chat + tool calls
- SDK ships openclaw.json / Nanobot config samples

Legend In/Out CPU NPU Agent Data UI

Use Case 6: VLA Model, Dual Robot Arms

Overview

With RK3588 as host and RK1828 as co-processor, an action-policy model runs on-device and fuses vision, language, and action (multimodal VLA): what the camera sees and what the operator says are turned by a single model into robot actions, which the host sends to the arms. The demo uses two arms (parallel-jaw gripper on the left, two-finger gripper on the right) to pick up and place plush toys and woven baskets.

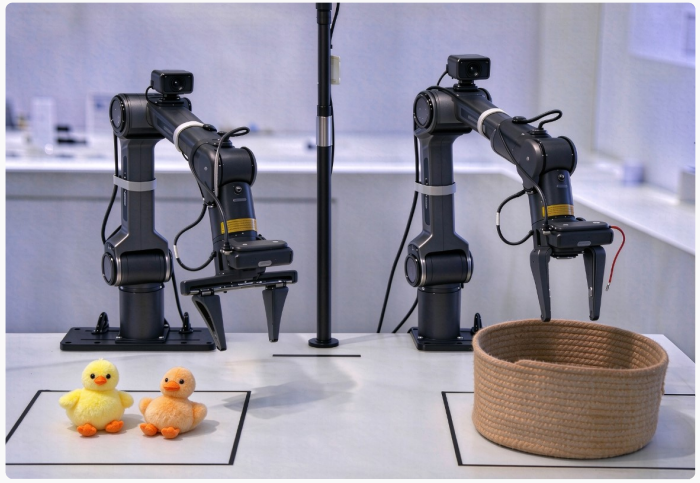
Benefits

It supports mainstream VLA models such as OpenPI, GR00T, LingBot, and MiniCPM, selectable per scenario and compute budget. Inference runs fully on-device, with no cloud, so it keeps working with the network down.

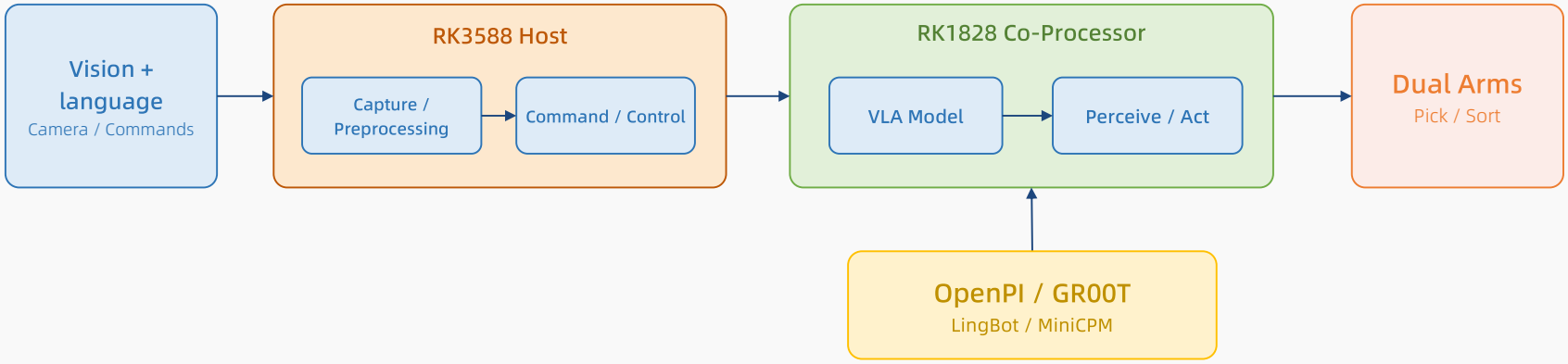
Use Cases

Ideal for embodied robots, smart picking, and industrial sorting: line loading and unloading, warehouse picking, booth demos, and flexible stations where the robot does what you say.

- Pick-and-Place Demo



Use Case 6: VLA Model, Dual Robot Arms



Performance:

- Vision-language-action in one model (VLA)
- Low-latency on-device inference, works without cloud
- Supports OpenPI / GR00T / LingBot / MiniCPM

Architecture:

- RK3588 + RK1828 pair; RK1828 accelerates multimodal inference
- VLA: part recognition / 3D pose estimation / command parsing / action sequence
- On the demo rig: two arms (parallel-jaw gripper left, two-finger gripper right)

Legend In/Out CPU NPU Agent Data UI

Use Case 7: Avatar + Local KB Q&A

Overview

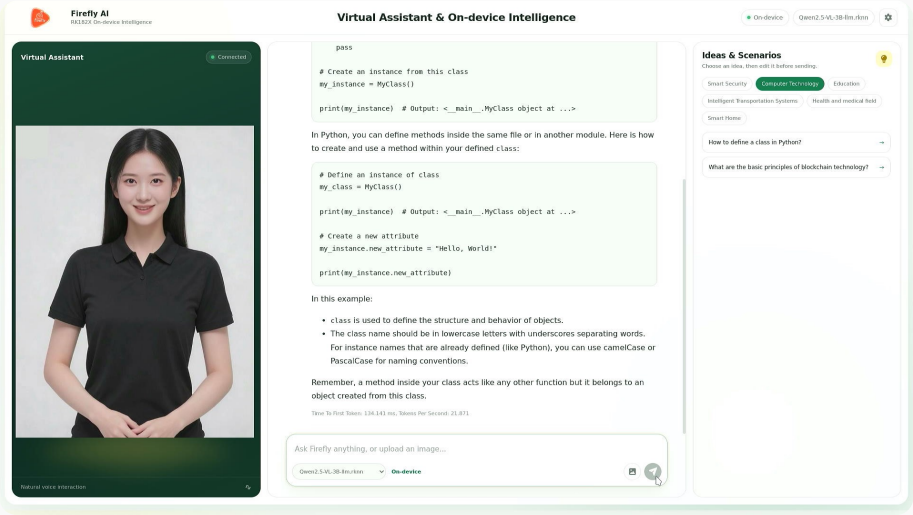
On RK3588 this turns text into live audio and video of an avatar: an external system pushes text over a Unix domain socket and the avatar speaks immediately, outputting 720 × 940 (RGB888) at 20 FPS with PCM audio at 16 kHz / 16-bit / mono.

The service starts from a desktop shortcut or with systemctl, so it can run unattended for months and be maintained remotely. Unlike a looping pre-rendered video, it can be interrupted and its layer can be hidden: users can stop the speech at any time and show or hide the avatar on demand, on a DSI or HDMI landscape display.

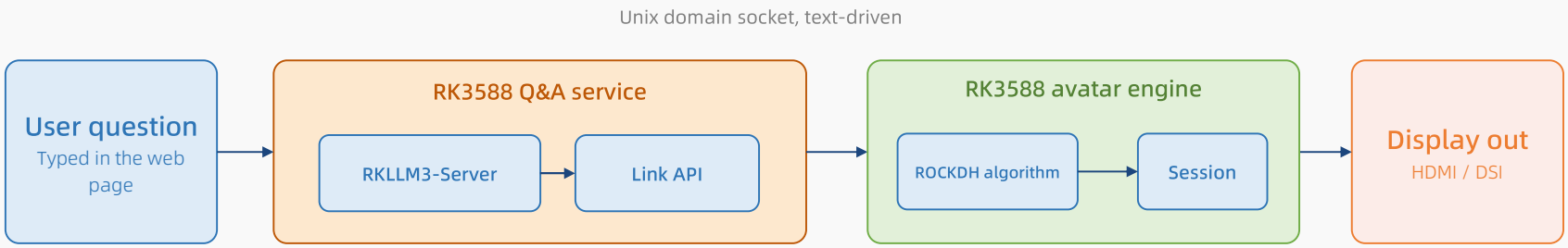
Use Cases

Ideal for showrooms, receptions, banks, hospitals, and guided tour announcements, plus virtual anchors driven by knowledge-base answers. Requires an RK3588 board with display and audio output, running Debian Linux.

- Avatar and on-device AI



Use Case 7: Avatar + Local KB Q&A



Performance:

- 720 × 940 (RGB888) at 20 FPS, rendered live;
- PCM 16 kHz / 16-bit / mono audio with lip-sync;
- Interruptible: speech stops immediately for a new command;
- Hideable: the avatar layer can be shown or hidden at runtime.

Architecture:

- Web Q&A answer text drives the avatar over a Unix domain socket;
- ROCKDH avatar algorithm;
- One-click desktop launch + systemctl start/stop;
- Q&A web service and avatar service deploy and start/stop independently.

Legend In/Out CPU NPU Agent Data UI

Note: some algorithms require Rockchip authorization

Use Case 8: Inspection + AI SOP Checks

Overview

A local inspection system on RK3588 + RK1828 using PatchCore unsupervised anomaly detection: build a baseline from normal samples only, then score new parts. Ideal when defects are rare and hard to collect samples for.

Benefits

Instead of a bare pass/fail verdict, the inspection page shows a heat map that locates the anomaly. You can run continuously, or pause at an anomaly and review it step by step – balancing throughput with human confirmation.
Traditional vision relies on thresholds or template matching, or needs many defect samples to train a classifier. Here only normal samples are needed: to switch to a different product or station, just collect a new batch of good parts.

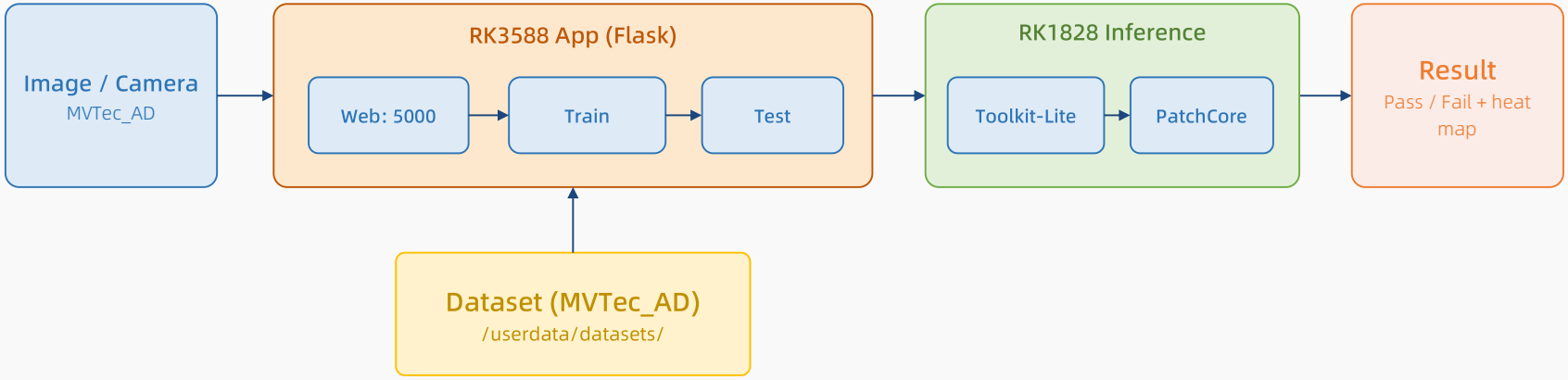
Use Cases

Ideal for in-line appearance checks, incoming inspection, and re-inspection at QC stations.

- Firefly Industrial Defect Inspection



Use Case 8: Inspection + AI SOP Checks



Performance:

- Training and inference both on board, no server;
- RKNN3-Toolkit-Lite runs .rknn on board;
- Flask on port 5000 for remote QC stations on the LAN;
- Launch from a desktop icon; the browser opens itself.

Architecture:

- Baseline from normal samples (PatchCore_Train);
- Inference returns an anomaly score + defect heat map (localization);
- Requires RKNN3.service running on board;
- The web page imports images and labels them OK/NG to build the reference set.

Legend In/Out CPU NPU Agent Data UI

Use Case 9: Smart Home Control

Overview

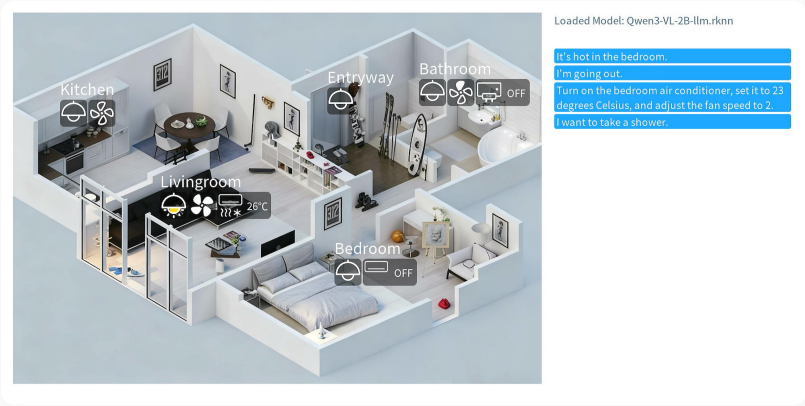
On a device with a screen and a microphone, this closes the offline voice loop: audio_detector wakes on sound, audio_input captures audio, asr recognizes speech offline, the local Nanobot Agent decides what to do, the MCP layer calls registered device tools over standard MCP, then tts speaks the reply and ui / chat_anim / loading_anim give LVGL animation feedback.

Unlike assistants that need the internet, wake, recognition, understanding and decisions all happen locally. The UI and local services run on ws_server and pyserver, and the sentry module can link person detection with voice interaction.

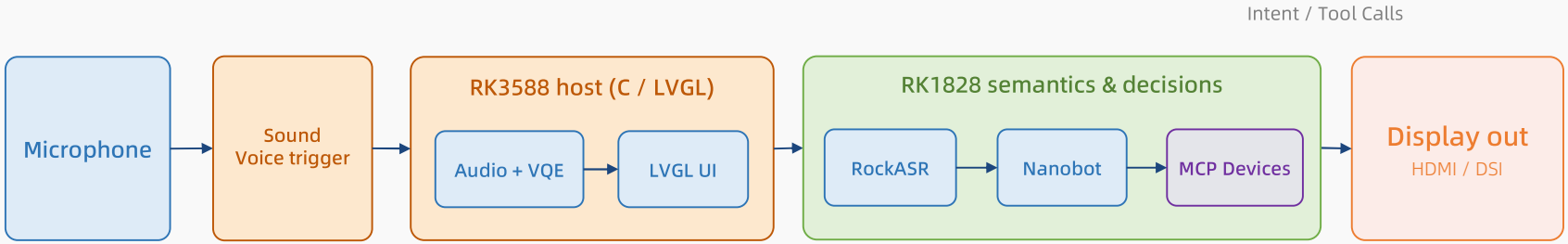
Use Cases

Ideal for voice access control, touch-screen voice terminals, and smart-home hubs where the controlled device is a home appliance.

- Smart Home AI Gateway



Use Case 9: Smart Home Control



Performance:

- Fully offline loop: wake, recognize, understand, decide - all local;
- Lightweight LVGL embedded UI;
- Register an MCP tool to add devices, no main-program change;
- The sentry module ties vision detection to voice interaction.

Architecture:

- Audio_detector VAD triggers on sound (no wake word);
- ASR transcribes, then the Nanobot Agent picks intent and tools;
- Standard MCP for access control / device control; RockTTS speaks locally;
- Controlled targets extend to home appliances via MCP tool registration.

Legend In/Out CPU NPU Agent Data UI

Note: some algorithms require Rockchip authorization

Use Case 10: Multi-Card + LLM + Local Agent

Overview

To run a large model on an edge device: the model is split across cards and runs distributed inference on several RK1828 cards, so 4-card combinations such as Qwen3.8-27B and Gemma-4-31B, and 2-card combinations such as Qwen3.5-9B and Gemma-4-12B, all run on-device.

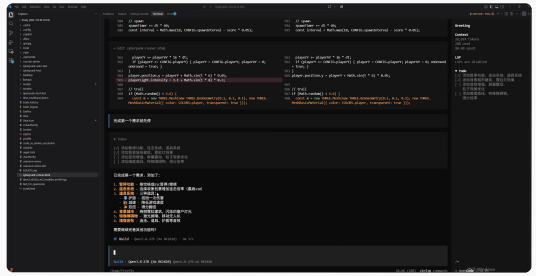
Instead of a fixed-model inference box, this is an orchestration service: any app using the OpenAI API can run chat and inference on the local network, with no internet.

Use Cases

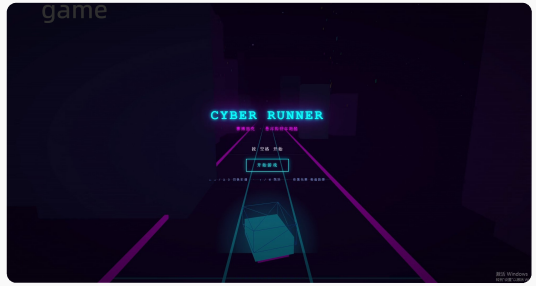
Ideal for private deployment of on-device LLMs, multi-card inference services, and OpenAI-compatible endpoints on a LAN. Requires several RK1828 cards and a model with a matching multi-card RKNN build (not every model splits automatically).

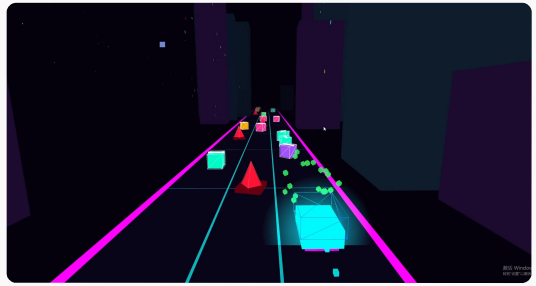
- Model: OpenCode



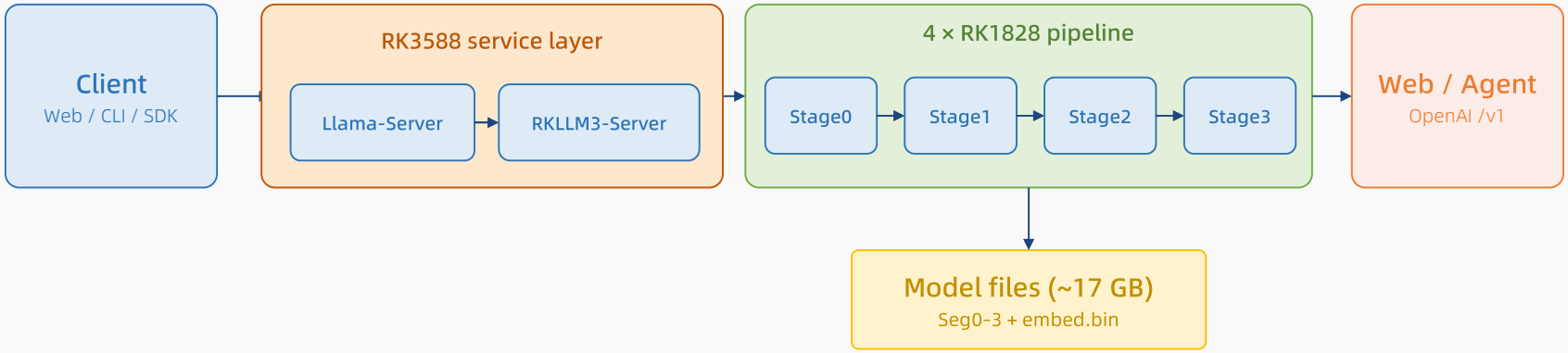


- Sample Output: Cyberpunk Parkour





Use Case 10: Multi-Card + LLM + Local Agent



Performance:

- Prefill 590.57 tok/s, Decode 13.04 tok/s (Qwen3.8-27B measured);
- Per stage: Prefill 634.11 tok/s, Decode 50.91 tok/s (pipeline overlap);
- Effective context 51,200 tokens; vocab 248,320; embedding dim 5,120;
- 4 cards auto-assigned Stage0-3.

Architecture:

- Chain: RKNN3.service → RKLLM3-Server → Llama-Server :18080 → Agent;
- Llama_Server_Startup auto-locates Seg0-Seg3 and the safetensors weights;
- Basic check: RKNN_Multicard_Demo (Model-Zoo cross-compiled);
- CLI tests and the web service cannot use the cards at the same time.

Legend In/Out CPU NPU Agent Data UI

T-CHIP GROUP

RK1828



Andy Sun

+86 180-2211-8569

global@t-firefly.com