Subsystems:

- **MPC ↔ GCE ↔ Proximity ↔ AVT** → Serial communication (tested OK).
- **CCD → Proximity → Monitor** → Raw video flow (tested OK).
- **CCD → AVT → Monitor** → Video flow (tested OK).
- **CCD → Proximity → AVT → Monitor → Fails in Debug Mode**, but **works in Flash mode**.

- **Processors & Tools:**
  - **AVT:** BF609 (Blackfin, CCES, video + serial).
  - **Proximity:** ARTIX-7 (Vivado, video + serial).
  - **GCE:** BF609 (serial).

So the problem only appears when you run under Debug mode in CCES (not when flashed).

Root Causes (Why it works in Flash, but fails in Debug)

This is a very common scenario in real-time embedded video/serial systems:

1. **Timing issues:**
   - In Debug mode, CCES adds overhead (breakpoints, JTAG control, step execution).
   - Video pipelines (especially with FPGA + DSP) require strict timing; debug interrupts break the frame sync → AVT never locks output.
2. **Cache / DMA setup not initialized properly in Debug:**
   - Sometimes flash bootloader configures cache, clocks, DDR init differently than debug loader.
   - In debug, you may be missing init of SDRAM controller, clocks, or PLL settings → Video data path breaks.
3. **Different loader (LDR) initialization:**
   - Flash boot uses initcode + startup assembly properly.
   - Debug bypasses parts of initcode (e.g., PLL, DDR, pin muxing) → FPGA interface may not see proper sync.

BLACKFIN

4. **JTAG halts DMA/video pipeline:**
    - On BF609, EMAC, PVP, and DMA engines stall in debug-halt state → downstream FPGA doesn't see continuous video.
    - That's why **direct CCD → AVT → Monitor works in debug**, but not with FPGA "Proximity" in the middle (needs continuous flow).

Step-by-Step Debugging / Fix Plan

1. **Check Initcode Differences:**
    - Compare your flash boot init sequence vs. debug startup (reset → load → run in CCES).
    - Ensure PLL, DDR, and pinmux setup is the same. If flash works but debug fails, your debug loader isn't setting them fully.
2. **Enable Continuous Run Mode in Debug:**
    - In CCES, go to **Debugger → Target Options → Run/Attach Mode**.
    - Avoid "Halt at main ()" → this stops DMA/video streams.
    - Instead, load + run without halting.
3. **Test with "Load Only" (No Halt):**
    - Use **Load Program** instead of full debug session.
    - This mimics flash execution while still letting you debug with printf ()/UART logs.
4. **Check FPGA (Proximity) sync signals:**
    - Use Vivado ILA or external LA/oscilloscope to see if FPGA is still receiving valid VSYNC/HSYNC/CLK when in Debug.
    - If sync collapses, then it's timing/jitter due to JTAG halt.
5. **Add Debug Logs Instead of Breakpoints:**
    - For video pipelines, never use step/run breakpoints.
    - Use UART logs (printf) or memory dump to check flow.
6. **Check Cache/DMA Config in Debug:**
    - Explicitly enable cache/DMA init in C runtime startup, not just in loader.
    - BF609 sometimes depends on LDR init for SDRAM setup → replicate in your startup file for debug.

BLACKFIN

Practical Fix You Can Try First

- In **CCES Debug Configurations**:
    - Disable “halt at main”.
    - Use “load program and run”.
    - If still failing, copy your init code from flash bootloader into the debug project’s startup.

🔹 **Option 1: CCES “Don’t Halt at Main, Load+Go”**

This is the quickest test to see if your debug problem is purely because JTAG halts the pipeline.

- I can give you step-by-step CCES settings (with screenshots/steps) so your code will behave almost like in Flash boot but still under Debug.
- Good if you want to test immediately without touching code.

🔹 **Option 2: Init Checklist (Flash vs Debug for BF609)**

This is more in-depth: we compare what’s initialized by the bootloader vs what debug skips.

- Covers PLL, DDR controller, pinmux, caches, DMA.
- Needed if your debug loader isn’t doing the same hardware init → then AVT video path won’t come up.
- Good if you want to fix root cause permanently and make debug environment stable.

CCES: Load & Run (avoid “halt at main”)

These steps assume CCES (Eclipse-based) and a Blackfin BF609 project. Menu names are the standard Eclipse/CCES names.

1. **Connect hardware & JTAG**
    - Ensure the target board is powered and JTAG cable is connected. Close other tools that may hold the JTAG.
2. **Open Debug Configurations**
    - In CCES: **Run → Debug Configurations…**

BLACKFIN

3. **Select or create your BF609 debug config**
    - In the left pane find your Blackfin/ADSP debug launch (the one you normally use).
    - If none exists: click the new launch configuration icon, choose your project and the ELF/application to load.
4. **Disable halting on startup**
    - In the configuration window look for the Startup tab (or the Debugger tab if there is no Startup tab).
    - Uncheck any option labelled like “Stop at main”, “Halt on startup”, “Halt target when connected”.
    - Look for and enable the option that says “Resume/Automatically resume after load” or “Download image and run” (wording varies slightly by CCES version).
    - Summary: *Load the image, then immediately resume the target instead of halting it.*
5. **Apply & Launch**
    - Click Apply, then Debug. The debugger should load your ELF and the processor should start running immediately (no halt at main).
6. **Verify no breakpoints will stop the pipeline**
    - Clear or disable any breakpoints that might be hit in video/serial paths (hardware breakpoints or software).
    - Avoid single-stepping while testing the video pipeline — stepping will disrupt DMA/video timings.
7. **If you need to inspect after it’s running: attach (non-invasive) if available**
    - If CCES supports *attach to running target*, use that option to connect to a running image after the pipeline is stable.
    - Otherwise, briefly pause (if necessary) only when safe — but avoid pauses during active frame streaming.

Quick test procedure (what to do after you Change the Debug Config)

1. Do the Load Run as above.
2. Immediately observe the monitor for AVT output (the path that previously failed in Debug).

BLACKFIN

3. Watch UART logs for boot/AVT messages.
4. If AVT output appears, you've confirmed JTAG halting was the problem.

If it still fails (quick troubleshooting tips)

- Switch to "Load Only / Run" (i.e., use *Run As → your application* or a "Load Program" option) so the target runs exactly as in Flash.
- Remove all breakpoints (they may be hit unexpectedly).
- Check startup/init differences — if Load +Run still fails, the debug loader may be skipping hardware init that flash sets (PLL, DDR, pinmux). If that happens, I'll prepare the full Flash vs Debug init checklist (PLL, DDR timing, pinmux, cache, DMA).
- Use UART printf logging instead of breakpoints to get runtime diagnostics without stopping the pipeline.
- If possible, use a Vivado ILA or logic analyzer on the FPGA input to see if the video clocks/sync stop when you connect the debugger.

CCES 2.11 – Debug Config "Load + Go" (Don't Halt at main)

1. Open Debug Configurations

- In the top menu: **Run → Debug Configurations…**
- This opens the **Debug Configurations** dialog.

---

2. Select Your Project Debug Launcher

- In the **left pane**, expand **"EZ-Kit Debuggers"** (or **CrossCore Debugger** depending on how you created it).
- You'll see entries like **[Your Project] Debug** or **[BF609] Session**.
- Select the one you normally use for JTAG debug.

---

3. Go to the Debugger Tab

- With your debug launcher selected → click on the **Debugger** tab (right side).
- You'll see **Target Options** and **Startup Options** sections.

BLACKFIN

4. Change Startup Options

Inside the **Startup Options** section:

- **Uncheck:**
    - **"Halt at program entry"** (this is the one that forces halt at main).
    - **"Halt target after load"** (sometimes just called *Halt after download*).
- **Check/Enable:**
    - **"Run immediately after load"** (this option makes the debugger load the ELF and resume execution without stopping).

So, the final state should be:

- "Halt at program entry" → **disabled**
- "Halt target after load" → **disabled**
- "Run immediately after load" → **enabled**

---

5. Apply and Save

- Click **Apply** then **Debug**.
- Now the program will be loaded into BF609 and **run immediately** without stopping at main ().

---

6. Verify Breakpoints

- Go to the **Breakpoints View** (Window → Show View → Breakpoints).
- Make sure no active breakpoints are set in critical ISR/DMA code.
- If any are active, disable or delete them before testing video flow.

---

7. Run & Observe

- Now test your pipeline: **CCD → Proximity → AVT → Monitor**.
- Since the debugger never halts the CPU or DMA engines, your video flow should behave just like Flash execution.

BLACKFIN

If you want to still attach after it's running → in **Debug Configurations → Debugger tab**, enable **"Attach to running target"** instead of reset-load. This way you can connect without disturbing the pipeline.

BLACKFIN