adi_jtag_dp_scan
ahbap_debugport_init
dap_ap_select
dap_setup_accessport
mem_ap_read_atomic_u32
mem_ap_read_buf_u16
mem_ap_read_buf_u32
mem_ap_read_buf_u8
mem_ap_read_u32
mem_ap_write_atomic_u32
mem_ap_write_u32

int ahbap_debugport_init (struct adiv5_dap * *dap*)

Initialize a DAP. This sets up the power domains, prepares the DP for further use, and arranges to use AP #0 for all AP operations until dap_ap-select() changes that policy.

Parameters:

dap The DAP being initialized.

Todo:

Rename this. We also need an initialization scheme which account for SWD transports not just JTAG; that will need to address differences in layering. (JTAG is useful without any debug target; but not SWD.) And this may not even use an AHB-AP ... e.g. DAP-Lite uses an APB-AP.

Definition at line 981 of file arm_adi_v5.c.

References adiv5_dap::ap_current, dap_ap_select(), dap_queue_dp_read(), dap_queue_dp_write(), and dap_run().

```
{
	uint32_t ctrlstat;
	int cnt = 0;
	int retval;

	LOG_DEBUG(" ");

	/* JTAG-DP or SWJ-DP, in JTAG mode
	 * ... for SWD mode this is patched as part
	 * of link switchover
	 */
	if (!dap->ops)
		dap->ops = &jtag_dp_ops;

	/* Default MEM-AP setup.
	 *
	 * REVISIT AP #0 may be an inappropriate default for this.
	 * Should we probe, or take a hint from the caller?
	 * Presumably we can ignore the possibility of multiple APs.
	 */
	dap->ap_current = !0;
	dap_ap_select(dap, 0);
```

```
	/* DP initialization */

	retval = dap_queue_dp_read(dap, DP_CTRL_STAT, NULL);
	if (retval != ERROR_OK)
		return retval;

	retval = dap_queue_dp_write(dap, DP_CTRL_STAT, SSTICKYERR);
	if (retval != ERROR_OK)
		return retval;

	retval = dap_queue_dp_read(dap, DP_CTRL_STAT, NULL);
	if (retval != ERROR_OK)
		return retval;

	dap->dp_ctrl_stat = CDBGPWRUPREQ | CSYSPWRUPREQ;
	retval = dap_queue_dp_write(dap, DP_CTRL_STAT, dap->dp_ctrl_stat);
	if (retval != ERROR_OK)
		return retval;

	retval = dap_queue_dp_read(dap, DP_CTRL_STAT, &ctrlstat);
	if (retval != ERROR_OK)
		return retval;
	if ((retval = dap_run(dap)) != ERROR_OK)
		return retval;

	/* Check that we have debug power domains activated */
	while (!(ctrlstat & CDBGPWRUPACK) && (cnt++ < 10))
	{
		LOG_DEBUG("DAP: wait CDBGPWRUPACK");
		retval = dap_queue_dp_read(dap, DP_CTRL_STAT, &ctrlstat);
		if (retval != ERROR_OK)
			return retval;
		if ((retval = dap_run(dap)) != ERROR_OK)
			return retval;
		alive_sleep(10);
	}

	while (!(ctrlstat & CSYSPWRUPACK) && (cnt++ < 10))
	{
		LOG_DEBUG("DAP: wait CSYSPWRUPACK");
		retval = dap_queue_dp_read(dap, DP_CTRL_STAT, &ctrlstat);
		if (retval != ERROR_OK)
			return retval;
		if ((retval = dap_run(dap)) != ERROR_OK)
			return retval;
		alive_sleep(10);
	}

	retval = dap_queue_dp_read(dap, DP_CTRL_STAT, NULL);
	if (retval != ERROR_OK)
		return retval;
	/* With debug power on we can activate OVERRUN checking */
	dap->dp_ctrl_stat = CDBGPWRUPREQ | CSYSPWRUPREQ | CORUNDETECT;
	retval = dap_queue_dp_write(dap, DP_CTRL_STAT, dap->dp_ctrl_stat);
	if (retval != ERROR_OK)
		return retval;
	retval = dap_queue_dp_read(dap, DP_CTRL_STAT, NULL);
	if (retval != ERROR_OK)
		return retval;

	return ERROR_OK;
}
```

Here is the call graph for this function: